
Improve your API with OpenAPI
Rob Allen
PHPUK, February 2022

APIs Power the Internet

APIs Power the Internet

API Descriptions Power APIs

The OpenAPI Specification (OAS) defines a standard, programming language-agnostic interface description for HTTP APIs, which allows both humans and computers to discover and understand the capabilities of a service

https://spec.openapis.org/oas/latest.html

It's about

documentation

It's about

design-first

It's about

communicating changes

It's about

development workflows

It's about

standardisation

It's about

a contract

"Using a consistent API description will help increase adoption of APIs across government by reducing time spent in understanding different APIs.

gov.uk


Anatomy of the specification

openapi.yaml

```yaml
openapi: "3.1.0"  # or "3.0.3"
info: # ...
servers: # ...
paths: # ...
webhooks: # ...
components: # ...
security: # ...
tags: # ...
externalDocs: # ...
```

Metadata

```yaml
info:
  title: Rock-Paper-Scissors
  version: "1.0.0"
  description: An implementation of Rock-Paper-Scissors.
  contact:
    name: "Rob Allen"
  license:
    name: The MIT License

servers:
  - url: https://rock-paper-scissors.example.com
    description: "RPS production API"
```

Endpoints

```
paths:
  /games:
    get:
      # ...
    post:
      # ...
  '/games/{game_id}/moves':
    post:
      # ...
  '/games/{game_id}/judgement':
    get:
      # ...
```

Endpoints

```yaml
paths:
  /games:
    post:
      operationId: createGame
      summary: Create a new game
      description: Create a new game of Rock-Paper-Scissors.
      requestBody:
        # ...
      responses:
        # ...
```

Endpoints

```yaml
paths:
  /games:
    post:
      operationId: createGame
      summary: Create a new game
      description: Create a new game of Rock-Paper-Scissors.
      requestBody:
        # ...
      responses:
        # ...
```

Endpoints

```yaml
paths:
  /games:
    post:
      operationId: createGame
      summary: Create a new game
      description: Create a new game of Rock-Paper-Scissors.
      requestBody:
        # ...
      responses:
        # ...
```

Endpoints

```yaml
paths:
  /games:
    post:
      operationId: createGame
      summary: Create a new game
      description: Create a new game of Rock-Paper-Scissors.
      requestBody:
        # ...
      responses:
        # ...
```

RequestBody

```yaml
requestBody:
  description: Game to add
  required: true
  content:
    application/json:
      schema:
        $ref: '#/components/schemas/NewGameRequest'
```

RequestBody

```yaml
requestBody:
  description: Game to add
  required: true
  content:
    application/json:
      schema:
        $ref: '#/components/schemas/NewGameRequest'
```

RequestBody

```yaml
requestBody:
  description: Game to add
  required: true
  content:
    application/json:
      schema:
        $ref: '#/components/schemas/NewGameRequest'
```

Reuse of objects

`$ref` allows us to define once & use in many places

```yaml
components:
  schemas:
    GameId:
      type: string
      format: "uuid"
      examples:
        - "2BC08389-885A-4322-80D0-EF0DE2D7CD37"
    Player:
      type: string
      example: "Lucy"
```

Reuse of objects

`$ref` allows us to define once & use in many places

```yaml
components:
  schemas:
    GameId:
      type: string
      format: "uuid"
      examples:
        - "2BC08389-885A-4322-80D0-EF0DE2D7CD37"
    Player:
      type: string
      example: "Lucy"
```

Reuse of objects

`$ref` allows us to define once & use in many places

```
components:
  schemas:
    GameId:
      type: string
      format: "uuid"
      examples:
        - "2BC08389-885A-4322-80D0-EF0DE2D7CD37"
    Player:
      type: string
      example: "Lucy"
```

Reuse of objects

`$ref` allows us to define once & use in many places

```yaml
components:
  schemas:
    GameId:
      type: string
      format: "uuid"
      examples:
        - "2BC08389-885A-4322-80D0-EF0DE2D7CD37"
    Player:
      type: string
      example: "Lucy"
```

Reuse of objects

`$ref` allows us to define once & use in many places

```yaml
components:
  schemas:
    GameId:
      type: string
      format: "uuid"
      examples:
        - "2BC08389-885A-4322-80D0-EF0DE2D7CD37"
    Player:
      type: string
      example: "Lucy"
```

Reuse of objects

`$ref` allows us to define once & use in many places

```yaml
components:
  schemas:
    GameId:
      type: string
      format: "uuid"
      examples:
        - "2BC08389-885A-4322-80D0-EF0DE2D7CD37"
    Player:
      type: string
      example: "Lucy"
```

Build on top of other components

```yaml
schemas:
  NewGameRequest:
    properties:
      player1:
        $ref: '#/components/schemas/Player'
      player2:
        $ref: '#/components/schemas/Player'
    required:
      - player1
      - player2
    examples:
      - '{"player1":"Lucy", "player2":"Dave"}'
```

Build on top of other components

```
schemas:
  NewGameRequest:
    properties:
      player1:
        $ref: '#/components/schemas/Player'
      player2:
        $ref: '#/components/schemas/Player'
    required:
      - player1
      - player2
    examples:
      - '{"player1":"Lucy", "player2":"Dave"}'
```

Build on top of other components

```yaml
schemas:
  NewGameRequest:
    properties:
      player1:
        $ref: '#/components/schemas/Player'
      player2:
        $ref: '#/components/schemas/Player'
    required:
      - player1
      - player2
    examples:
      - '{"player1":"Lucy", "player2":"Dave"}'
```

Build on top of other components

```
schemas:
  NewGameRequest:
    properties:
      player1:
        $ref: '#/components/schemas/Player'
      player2:
        $ref: '#/components/schemas/Player'
    required:
      - player1
      - player2
    examples:
      - '{"player1":"Lucy", "player2":"Dave"}'
```

Build on top of other components

```
schemas:
  NewGameRequest:
    properties:
      player1:
        $ref: '#/components/schemas/Player'
      player2:
        $ref: '#/components/schemas/Player'
    required:
      - player1
      - player2
    examples:
      - '{"player1":"Lucy", "player2":"Dave"}'
```

RequestBody

```yaml
requestBody:
  description: Game to add
  required: true
  content:
    application/json:
      schema:
        $ref: '#/components/schemas/NewGameRequest'
```

Responses

```yaml
responses:
  '201':
    $ref: '#/components/responses/NewGameResponse'
  '400':
    $ref: '#/components/responses/NewGameError'
  '500':
    $ref: '#/components/responses/InternalServerError'
```

Responses

```
responses:
  '201':
    $ref: '#/components/responses/NewGameResponse'
  '400':
    $ref: '#/components/responses/NewGameError'
  '500':
    $ref: '#/components/responses/InternalServerError'
```

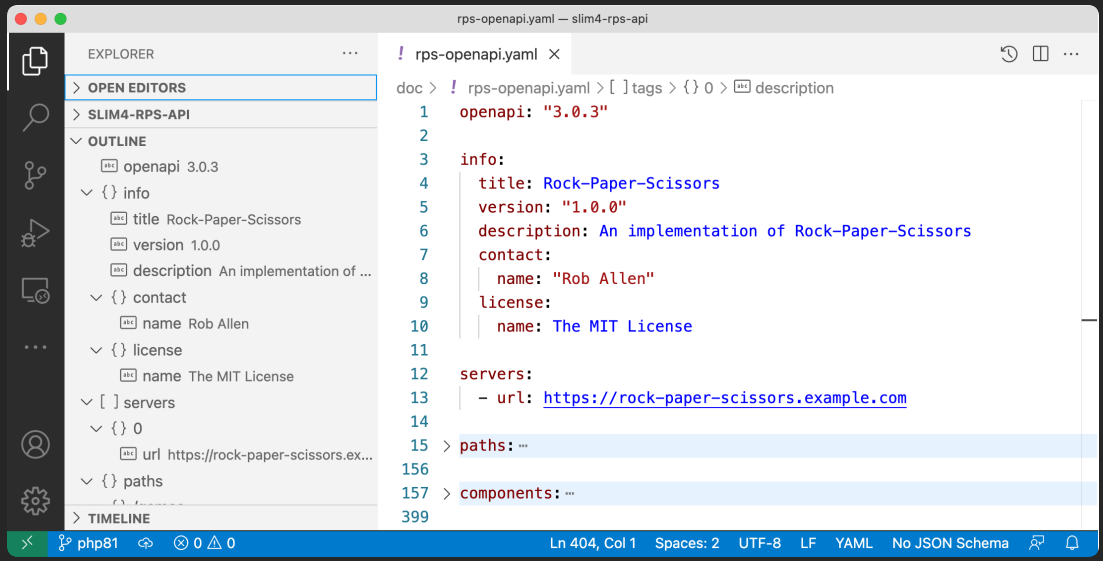
Responses

```
responses:
  '201':
    $ref: '#/components/responses/NewGameResponse'
  '400':
    $ref: '#/components/responses/NewGameError'
  '500':
    $ref: '#/components/responses/InternalServerError'
```

Writing your spec

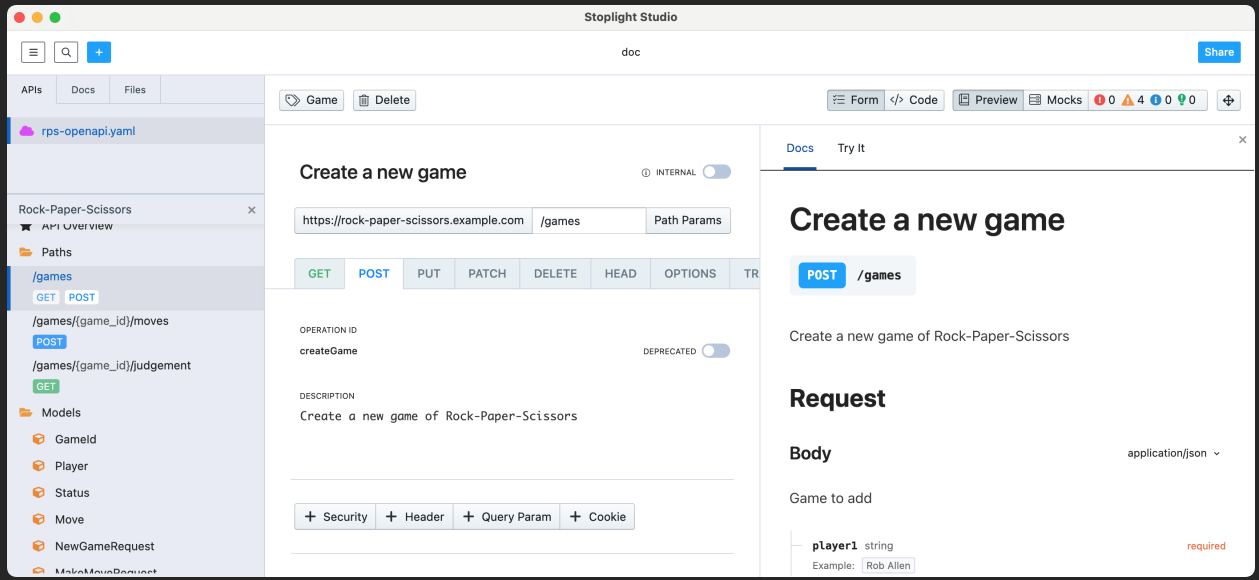
Editing

It's just text!



Editing

GUI tools: Stoplight, OpenAPI-GUI, Swagger Editor



Linting & validation

CLI tools: Spectral, openapi-spec-validator, etc.

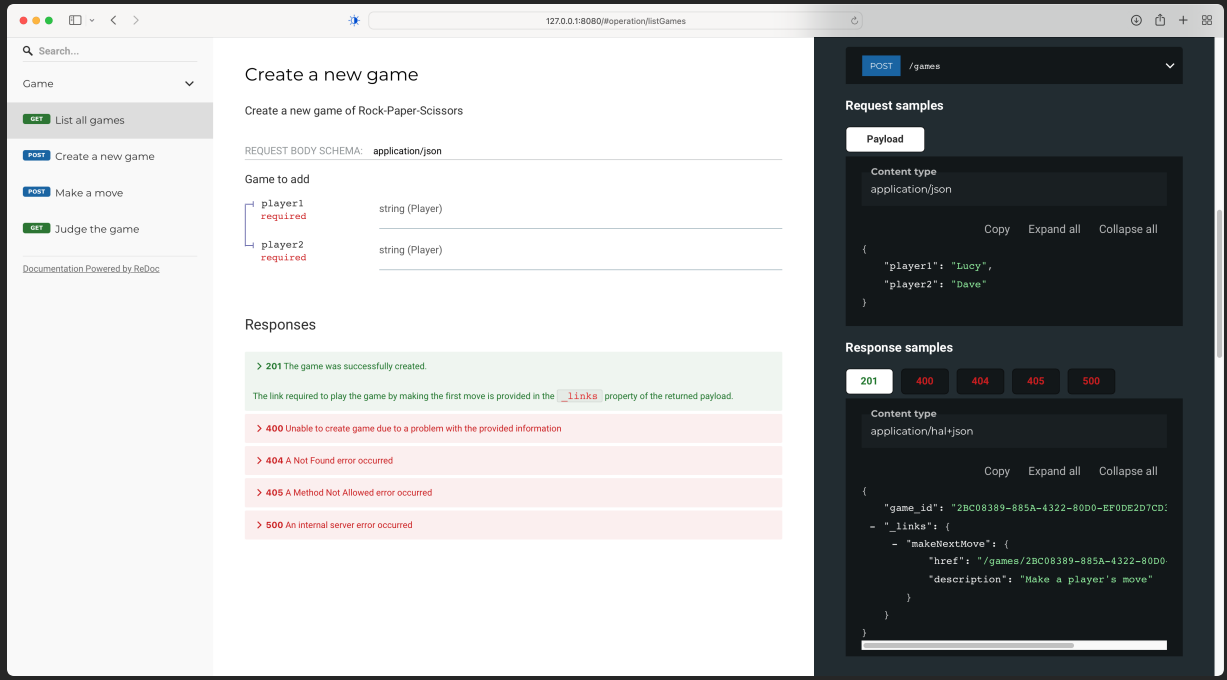
```
$ spectral lint rps-openapi.yaml
No results with a severity of 'error' or higher found!
```

Validation error

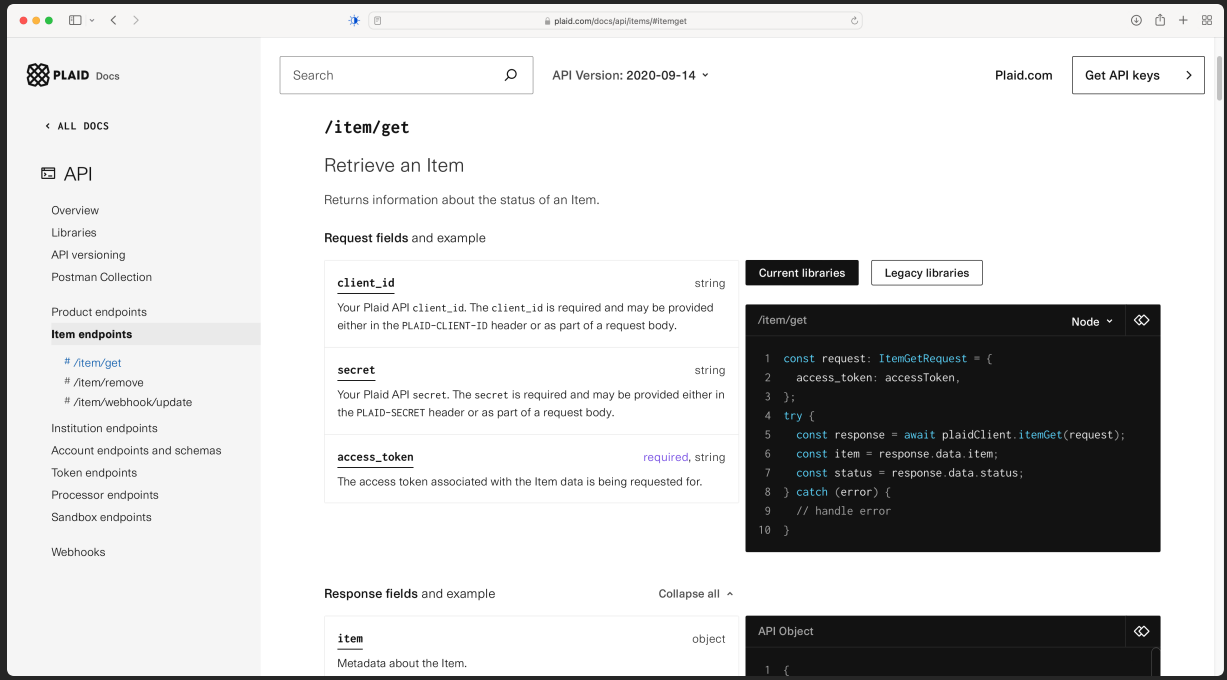
```
$ spectral lint rps-openapi.yaml

.../slim4-rps-api/doc/rps-openapi.yaml
3:6  warning  info-contact  Info object must have
  "contact" object.  info

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

Docs

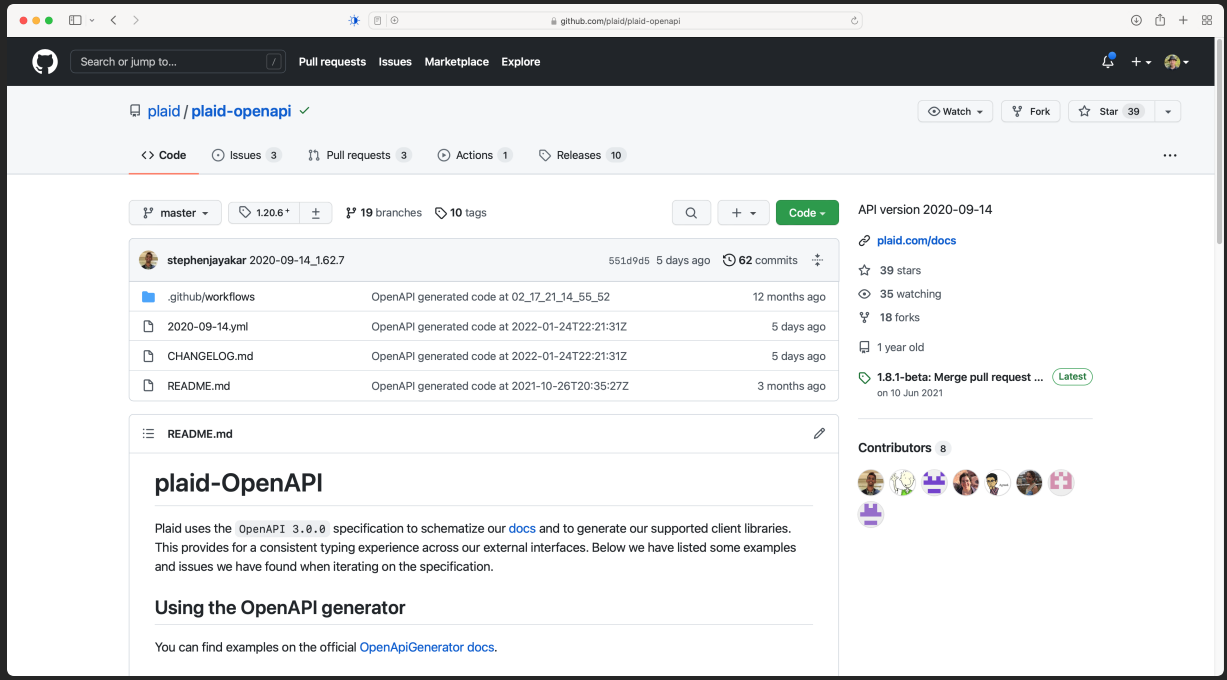
Docs


127.0.0.1:8080/#operation/listGames
Search...
Game
GET List all games
POST Create a new game
POST Make a move
GET Judge the game
Documentation Powered by ReDoc
Create a new game
Create a new game of Rock-Paper-Scissors
REQUEST BODY SCHEMA: application/json
Game to add
player1
required
string (Player)
player2
required
string (Player)
Responses
201 The game was successfully created.
The link required to play the game by making the first move is provided in the _links property of the returned payload.
400 Unable to create game due to a problem with the provided information
404 A Not Found error occurred
405 A Method Not Allowed error occurred
500 An internal server error occurred
POST /games
Request samples
Payload
Content type
application/json
Copy Expand all Collapse all
{
"player1": "Lucy",
"player2": "Dave"
}
Response samples
201 400 404 405 500
Content type
application/hal+json
Copy Expand all Collapse all
{
"game_id": "2BC08389-885A-4322-80D0-EF0DE2D7CD
"_links": {
"makeNextMove": {
"href": "/games/2BC08389-885A-4322-80D0
"description": "Make a player's move"
}
}
}

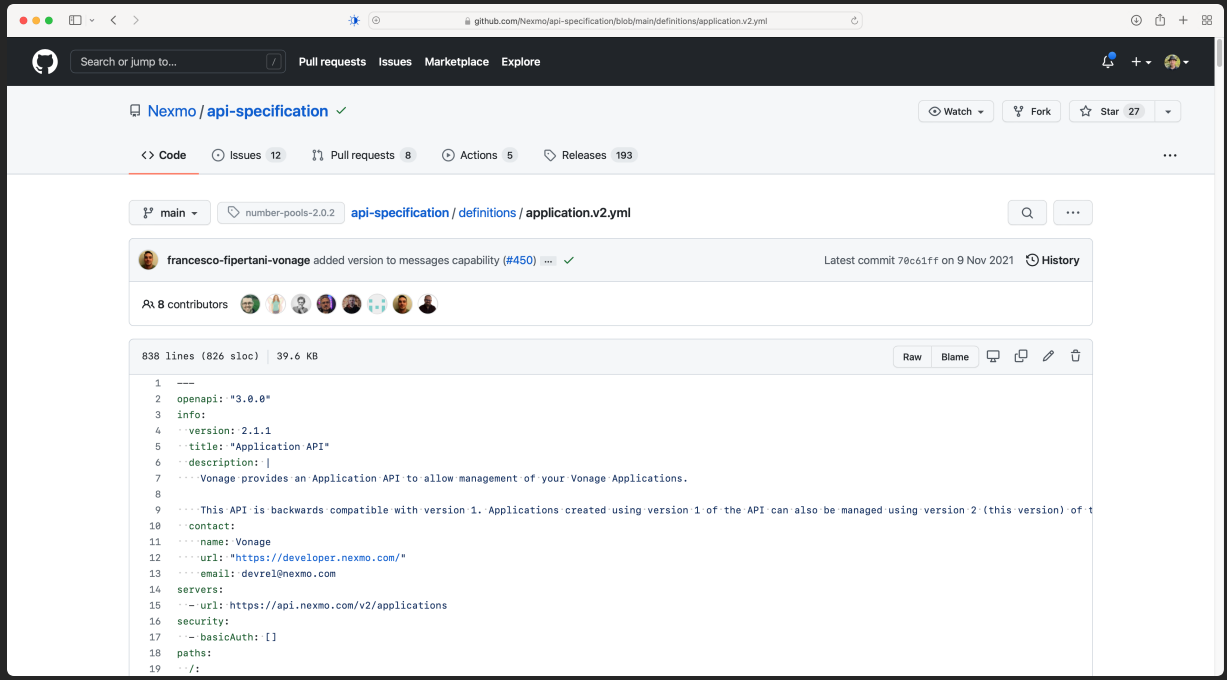
Docs


plaid.com/docs/api/items/#itemget
PLAID Docs
Search
API Version: 2020-09-14
Plaid.com
Get API keys
ALL DOCS
API
Overview
Libraries
API versioning
Postman Collection
Product endpoints
Item endpoints
/item/get
/item/remove
/item/webhook/update
Institution endpoints
Account endpoints and schemas
Token endpoints
Processor endpoints
Sandbox endpoints
Webhooks
/item/get
Retrieve an Item
Returns information about the status of an Item.
Request fields and example
client_id
string
Your Plaid API client_id. The client_id is required and may be provided either in the PLAID-CLIENT-ID header or as part of a request body.
secret
string
Your Plaid API secret. The secret is required and may be provided either in the PLAID-SECRET header or as part of a request body.
access_token
required, string
The access token associated with the Item data is being requested for.
Current libraries
Legacy libraries
/item/get
Node
const request: ItemGetRequest = {
access_token: accessToken,
};
try {
const response = await plaidClient.itemGet(request);
const item = response.data.item;
const status = response.data.status;
} catch (error) {
// handle error
}
Response fields and example
Collapse all
item
object
Metadata about the Item.
API Object
{

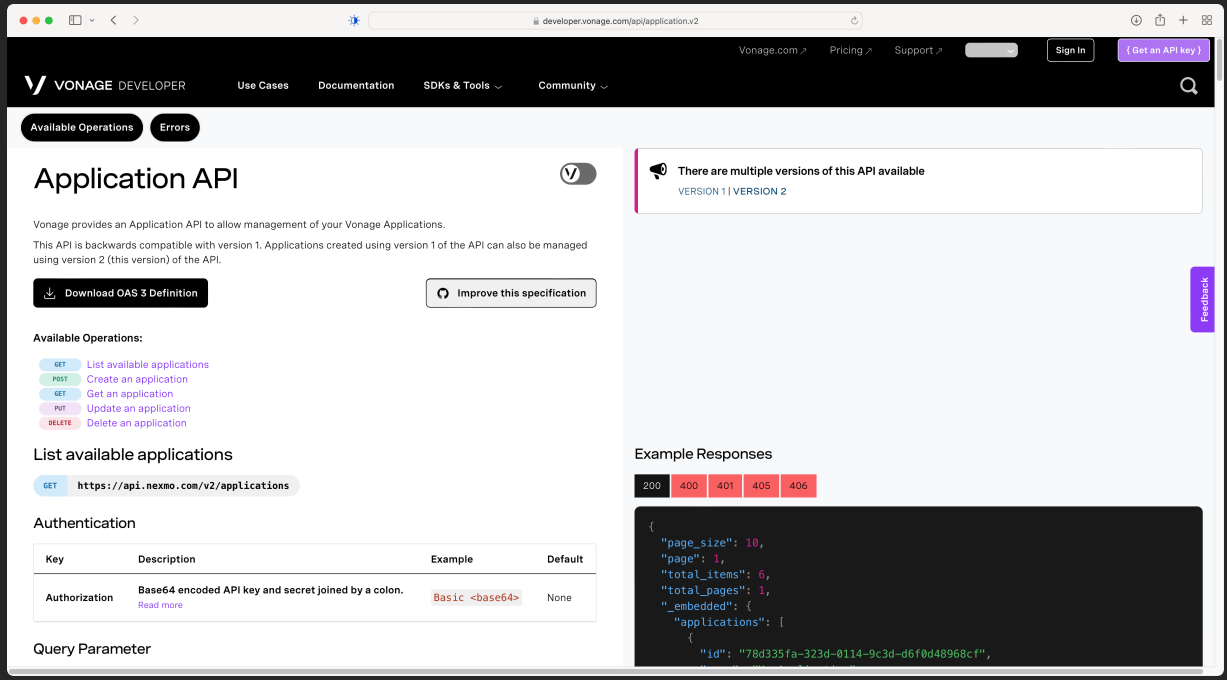
Docs


github.com/plaid/plaid-openapi
Search or jump to...
Pull requests
Issues
Marketplace
Explore
plaid / plaid-openapi
Watch
Fork
Star 39
Code
Issues 3
Pull requests 3
Actions 1
Releases 10
master
1.20.6
19 branches
10 tags
Code
stephenjayakar 2020-09-14_1.62.7
551d9d5 5 days ago
62 commits
.github/workflows OpenAPI generated code at 02_17_21_14_55_52 12 months ago
2020-09-14.yml OpenAPI generated code at 2022-01-24T22:21:31Z 5 days ago
CHANGELOG.md OpenAPI generated code at 2022-01-24T22:21:31Z 5 days ago
README.md OpenAPI generated code at 2021-10-26T20:35:27Z 3 months ago
README.md
plaid-OpenAPI
Plaid uses the OpenAPI 3.0.0 specification to schematize our docs and to generate our supported client libraries. This provides for a consistent typing experience across our external interfaces. Below we have listed some examples and issues we have found when iterating on the specification.
Using the OpenAPI generator
You can find examples on the official OpenApiGenerator docs.
API version 2020-09-14
plaid.com/docs
39 stars
35 watching
18 forks
1 year old
1.8.1-beta: Merge pull request ...
Latest
on 10 Jun 2021
Contributors 8

Docs


github.com/Nexmo/api-specification/blob/main/definitions/application.v2.yml
Search or jump to...
Pull requests
Issues
Marketplace
Explore
Nexmo / api-specification
Watch
Fork
Star 27
Code
Issues 12
Pull requests 8
Actions 5
Releases 193
main
number-pools-2.0.2
api-specification / definitions / application.v2.yml
francesco-fipertani-vonage added version to messages capability (#450)
Latest commit 70c61ff on 9 Nov 2021
History
8 contributors
838 lines (826 sloc) | 39.6 KB
Raw
Blame
---
openapi: "3.0.0"
info:
version: 2.1.1
title: "Application API"
description: |
Vonage provides an Application API to allow management of your Vonage Applications.
This API is backwards compatible with version 1. Applications created using version 1 of the API can also be managed using version 2 (this version) of t
contact:
name: Vonage
url: "https://developer.nexmo.com/"
email: devrel@nexmo.com
servers:
- url: https://api.nexmo.com/v2/applications
security:
- basicAuth: []
paths:
/:

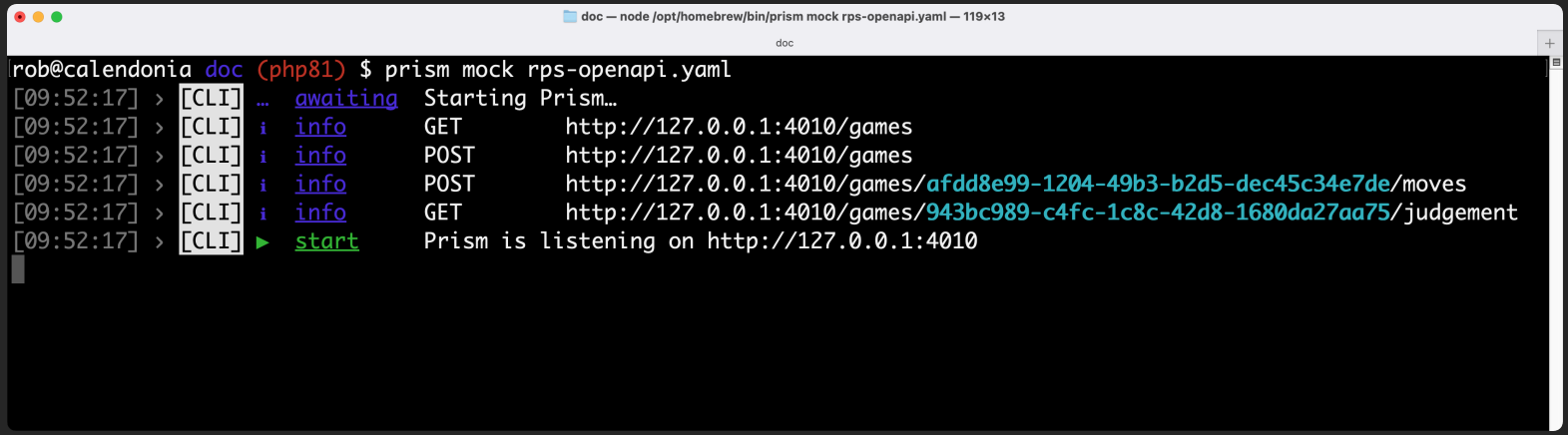
Docs


developer.vonage.com/api/application.v2
Vonage.com ↗
Pricing ↗
Support ↗
Sign In
{ Get an API key }
VONAGE DEVELOPER
Use Cases
Documentation
SDKs & Tools
Community
Available Operations
Errors
Application API
Vonage provides an Application API to allow management of your Vonage Applications.
This API is backwards compatible with version 1. Applications created using version 1 of the API can also be managed using version 2 (this version) of the API.
Download OAS 3 Definition
Improve this specification
There are multiple versions of this API available
VERSION 1 | VERSION 2
Feedback
Available Operations:
GET List available applications
POST Create an application
GET Get an application
PUT Update an application
DELETE Delete an application
List available applications
GET https://api.nexmo.com/v2/applications
Authentication
Key
Description
Example
Default
Authorization
Base64 encoded API key and secret joined by a colon.
Read more
Basic <base64>
None
Query Parameter
Example Responses
200 400 401 405 406
{
"page_size": 10,
"page": 1,
"total_items": 6,
"total_pages": 1,
"_embedded": {
"applications": [
{
"id": "78d335fa-323d-0114-9c3d-d6f0d48968cf",


Developers

Mock server

```
$ prism mock rps-openapi.yaml
```



Make API calls

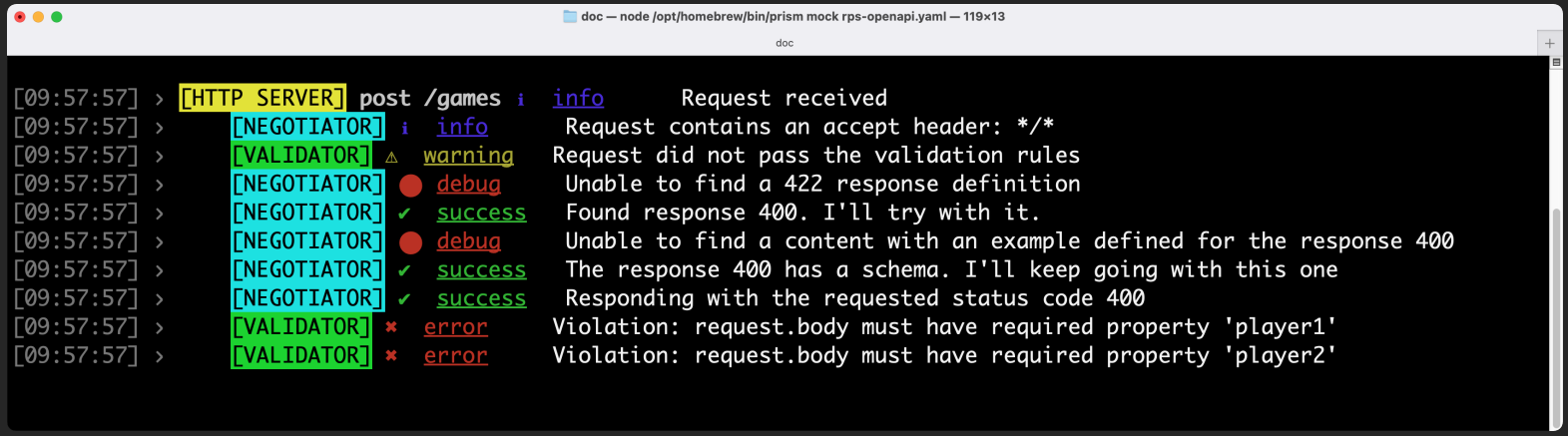
```
$ curl http://127.0.0.1:4010/games -d '{}'
```

Make API calls

```
$ curl http://127.0.0.1:4010/games -d '{}'
{"message":"Must provide both player1 and player2"}
```

Make API calls

```
$ curl http://127.0.0.1:4010/games -d '{}'
{"message":"Must provide both player1 and player2"}
```



Validation

The `schema` section can be used to validate the request *and* response

- Validate early and return a 422
- Validate that we return what we say we will
- Put it in CI to prevent regressions

But I already have validation!

Your code:

- isn't good enough!
- isn't reusable!
- doesn't match the docs!

But I already have validation!

Your code:

- isn't good enough!
- isn't reusable!
- doesn't match the docs!

However…

Business logic validation still needed!

Validation in PHP

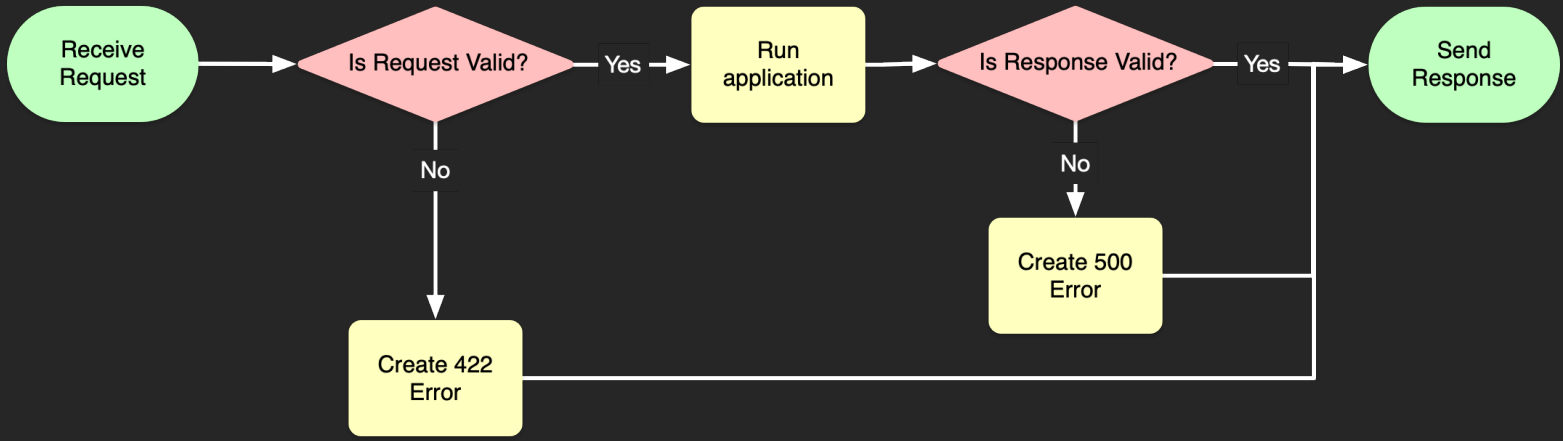
OpenAPI 3.0:

`league/openapi-psr7-validator`

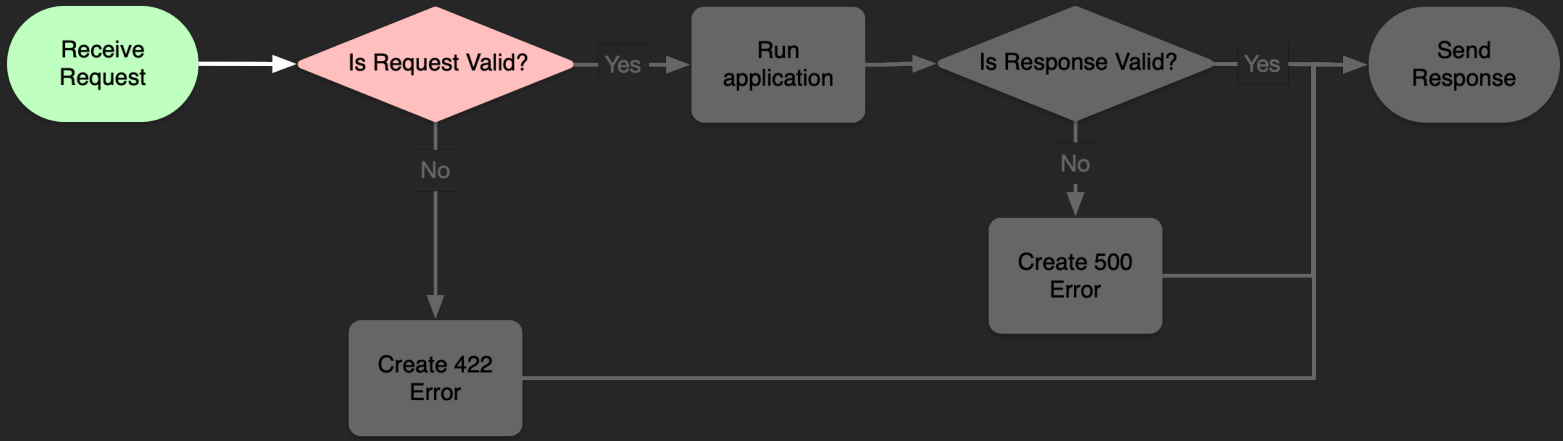
OpenAPI 3.1:

`opis/json-schema`

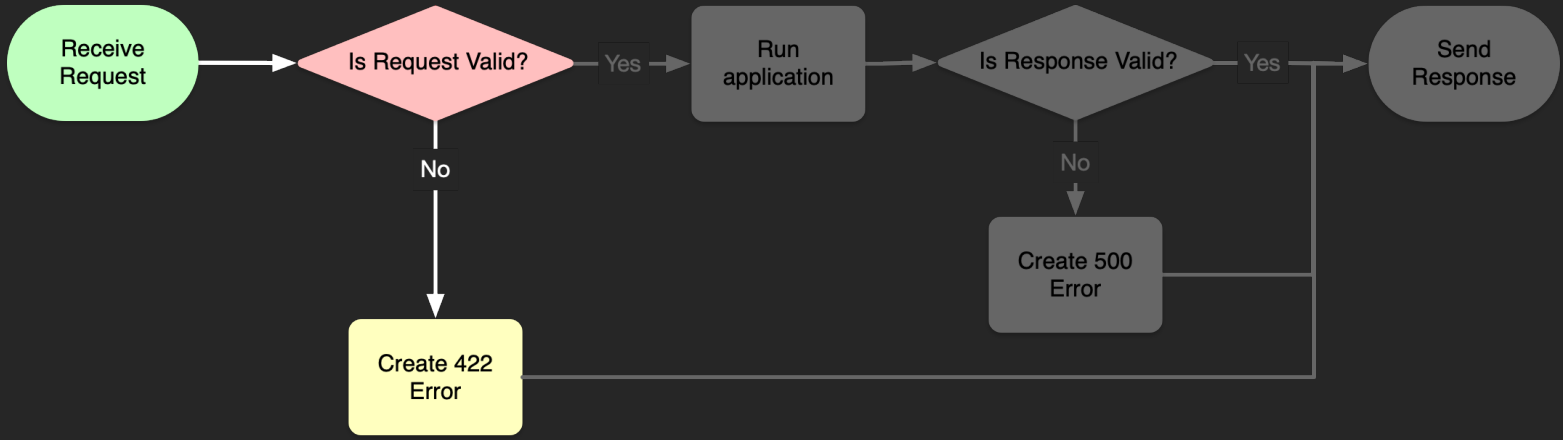
Validation middleware



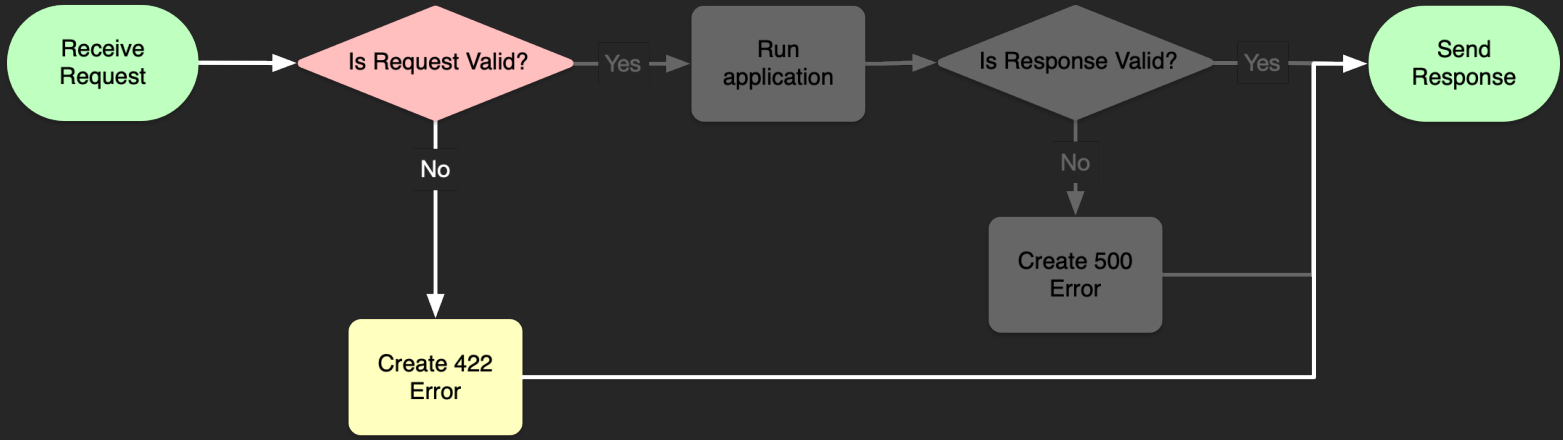
Test Request



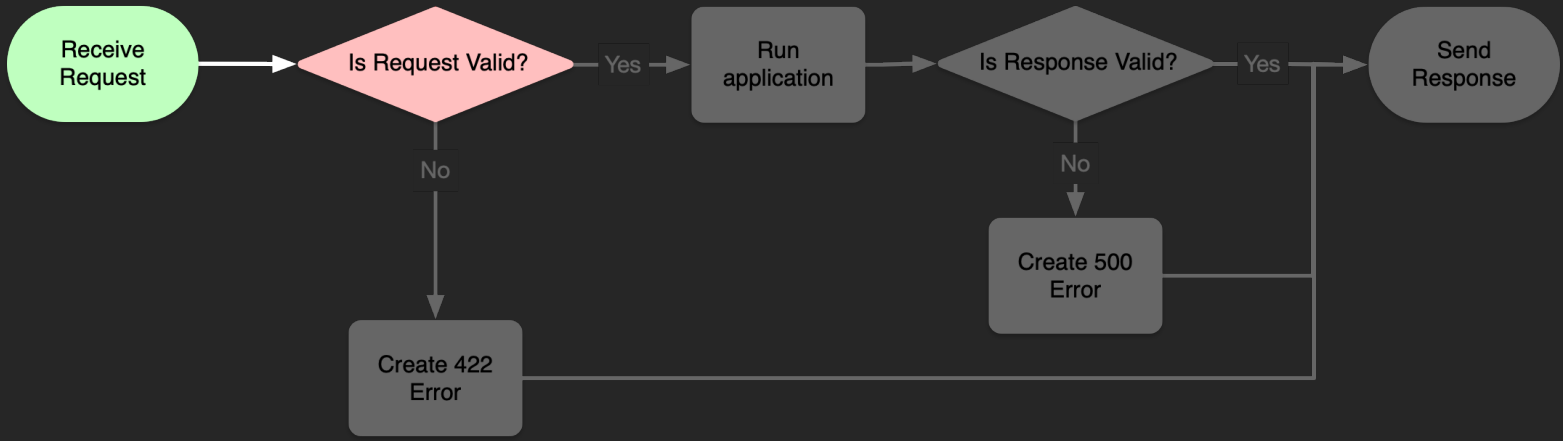
Request is invalid



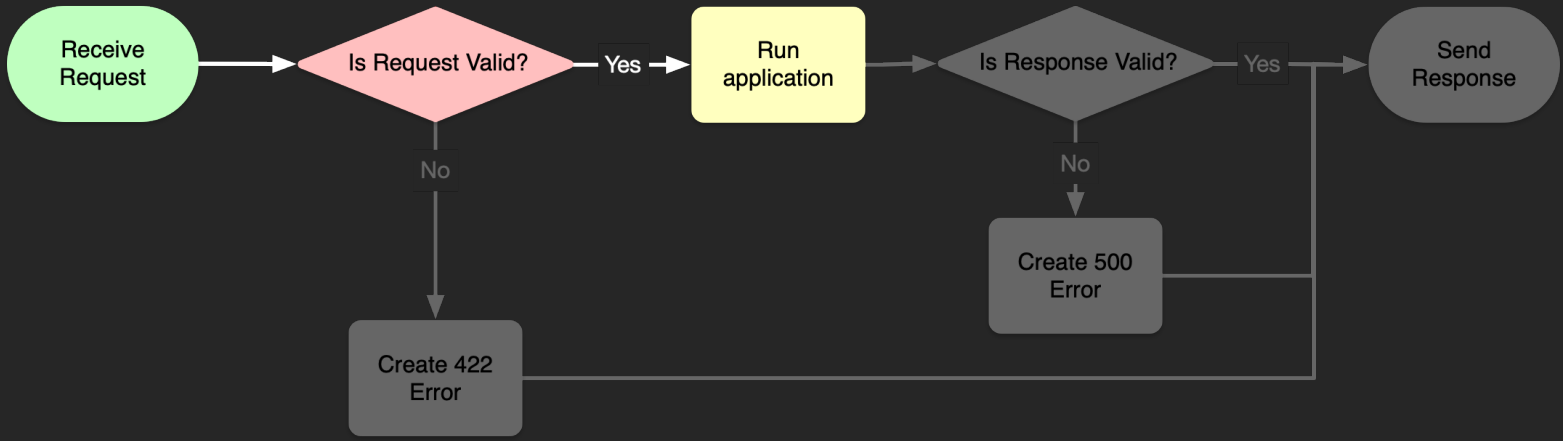
Request is invalid



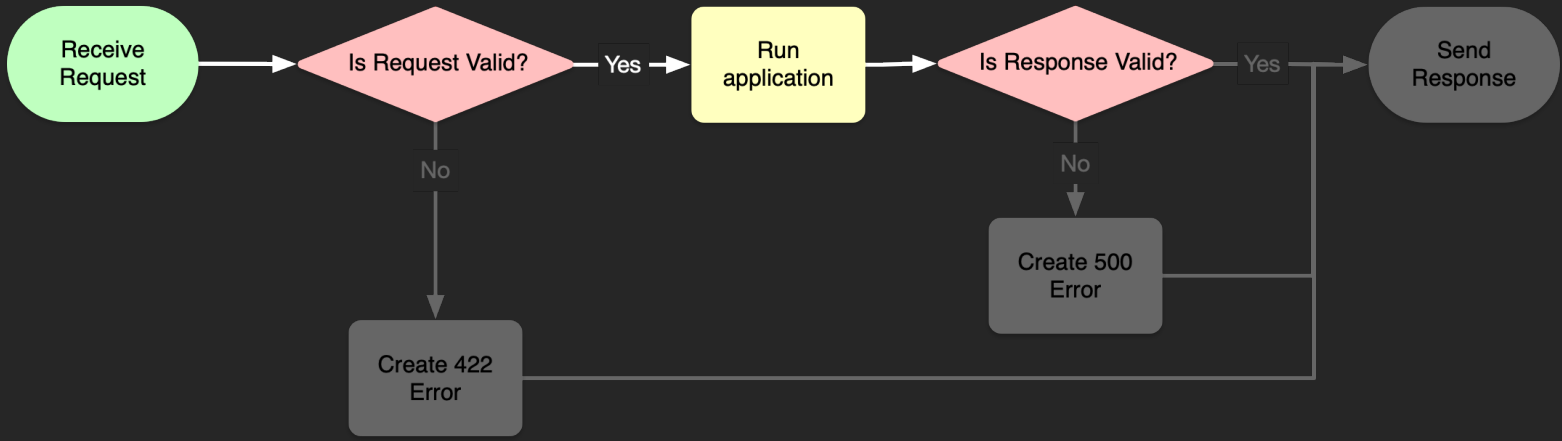
Test Request



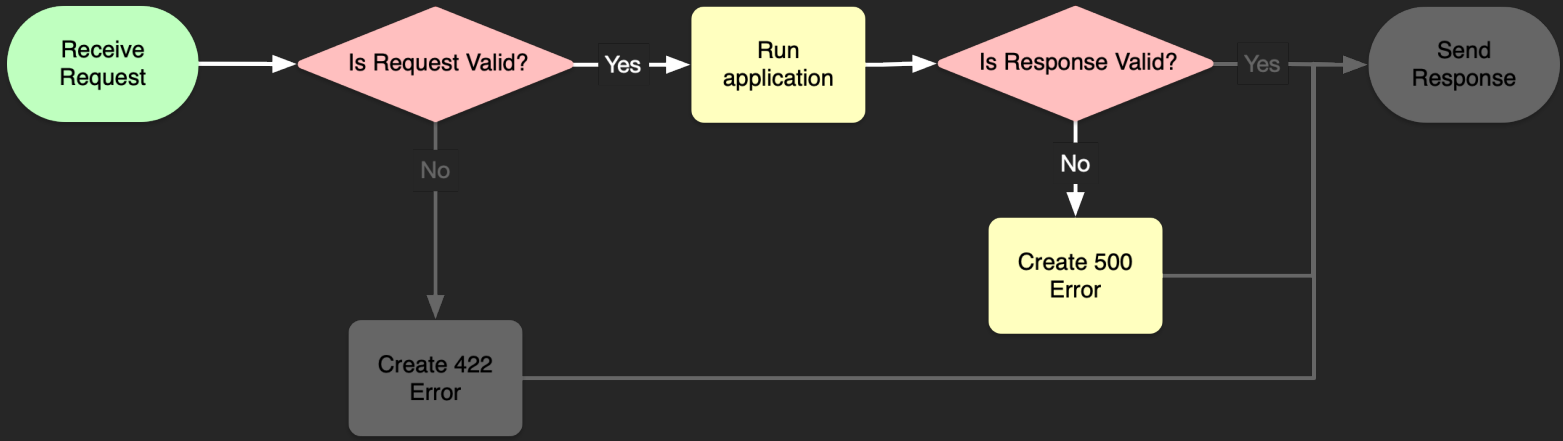
Request is valid



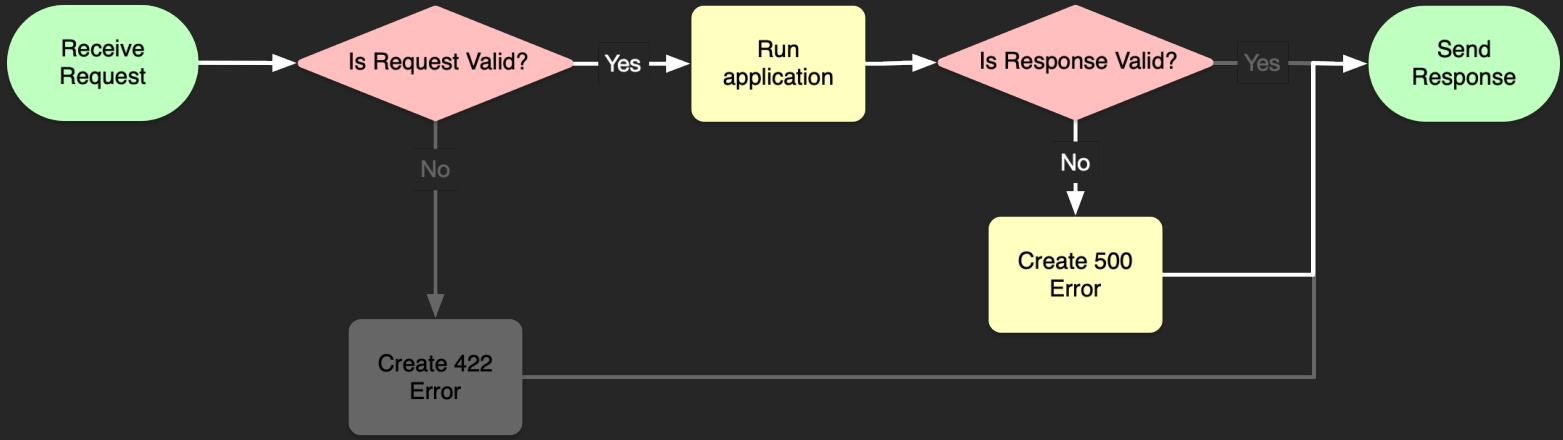
Test Response



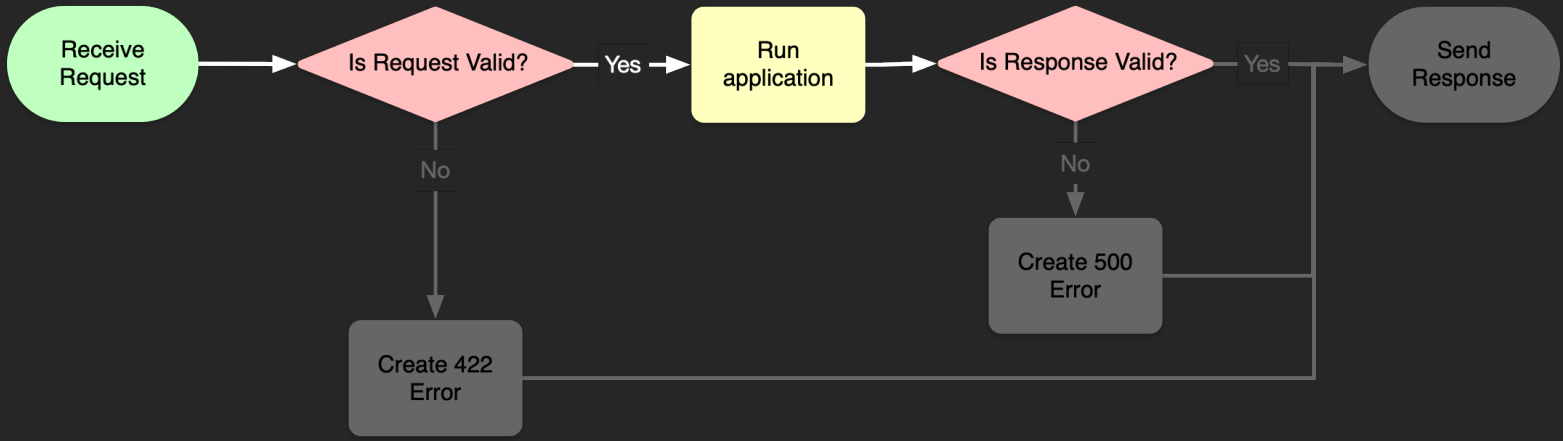
Response is invalid



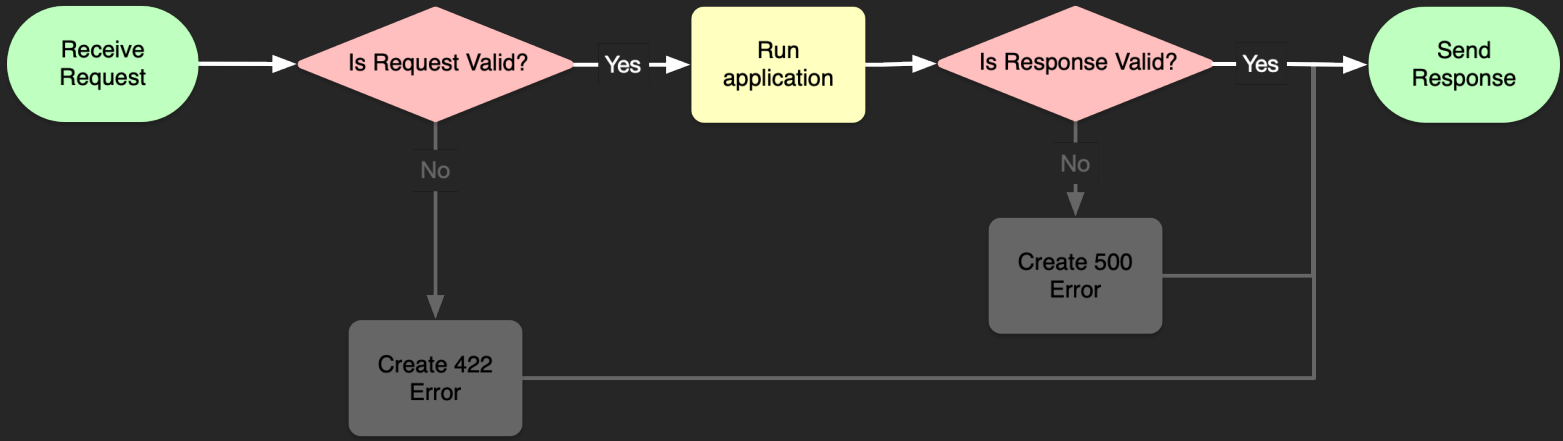
Response is invalid



Successful validation



Successful validation



Validation middleware

```php
class OpenApiValidationMiddleware implements MiddlewareInterface
{
  public function __construct(string $oasFilename, Cache $cache)
  {
    $builder = new ValidatorBuilder();
    $builder->fromYamlFile($oasFilename);
    $builder->setCache($cache)->overrideCacheKey('openapi');

    $this->reqValidator = $builder->getServerRequestValidator();
    $this->respValidator = $builder->getResponseValidator();
  }

  public function process($request, $handler)
  {
    try {
      // validate request
      $match = $this->reqValidator->validate($request);
    } catch (ValidationFailed $e) {
      throw new HttpException($request, 422, $e);
    }

    // process
    $response = $handler->handle($request);

    try {
      // validate response
      $this->respValidator->validate($match, $response);
      return $response;
    } catch (ValidationFailed $e) {
      throw new HttpException($request, 500, $e);
    }
  }
}
```

Validation middleware

```
public function __construct(string $oasFilename, Cache $cache)
{
  $builder = new ValidatorBuilder();
  $builder->fromYamlFile($oasFilename);
  $builder->setCache($cache)->overrideCacheKey('openapi');

  $this->reqValidator = $builder->getServerRequestValidator();
  $this->respValidator = $builder->getResponseValidator();
}
```

Validation middleware

```
public function __construct(string $oasFilename, Cache $cache)
{
  $builder = new ValidatorBuilder();
  $builder->fromYamlFile($oasFilename);
  $builder->setCache($cache)->overrideCacheKey('openapi');

  $this->reqValidator = $builder->getServerRequestValidator();
  $this->respValidator = $builder->getResponseValidator();
}
```

Validation middleware

```
public function __construct(string $oasFilename, Cache $cache)
{
  $builder = new ValidatorBuilder();
  $builder->fromYamlFile($oasFilename);
  $builder->setCache($cache)->overrideCacheKey('openapi');

  $this->reqValidator = $builder->getServerRequestValidator();
  $this->respValidator = $builder->getResponseValidator();
}
```

Validation middleware

```
public function __construct(string $oasFilename, Cache $cache)
{
  $builder = new ValidatorBuilder();
  $builder->fromYamlFile($oasFilename);
  $builder->setCache($cache)->overrideCacheKey('openapi');

  $this->reqValidator = $builder->getServerRequestValidator();
  $this->respValidator = $builder->getResponseValidator();
}
```

Validation middleware

```
public function process($request, $handler)
{
  try {
    // validate request
    $match = $this->reqValidator->validate($request);
  } catch (ValidationFailed $e) {
    throw new HttpException($request, 422, $e);
  }
```

Validation middleware

```
public function process($request, $handler)
{
  try {
    // validate request
    $match = $this->reqValidator->validate($request);
  } catch (ValidationFailed $e) {
    throw new HttpException($request, 422, $e);
  }
```

Validation middleware

```
public function process($request, $handler)
{
  try {
    // validate request
    $match = $this->reqValidator->validate($request);
  } catch (ValidationFailed $e) {
    throw new HttpException($request, 422, $e);
  }
```

Validation middleware

```
public function process($request, $handler)
{
  ...

  // process
  $response = $handler->handle($request);
```

Validation middleware

```
public function process($request, $handler)
{
  ...

  try {
    // validate response
    $this->respValidator->validate($match, $response);
    return $response;
  } catch (ValidationFailed $e) {
    throw new HttpException($request, 500, $e);
  }
}
```

Validation middleware

```
public function process($request, $handler)
{
  ...

  try {
    // validate response
    $this->respValidator->validate($match, $response);
    return $response;
  } catch (ValidationFailed $e) {
    throw new HttpException($request, 500, $e);
  }
}
```

Validation middleware

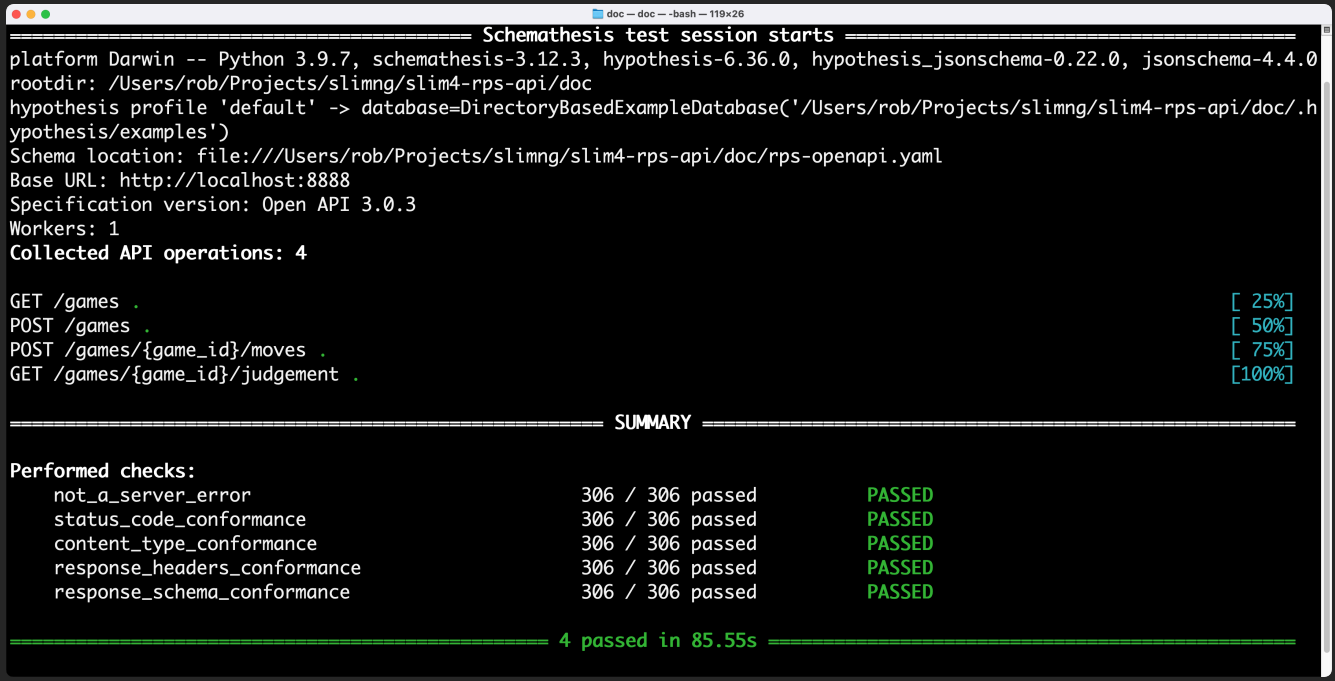
```
public function process($request, $handler)
{
  ...

  try {
    // validate response
    $this->respValidator->validate($match, $response);
    return $response;
  } catch (ValidationFailed $e) {
    throw new HttpException($request, 500, $e);
  }
}
```

Compliance Testing

Schemathesis reads your OpenAPI spec and tests your API against it

```
pip install schemathesis

schemathesis run --stateful=links --checks all \
  --base-url=http://localhost:8888 \
  doc/rps-openapi.yaml
```

Compliance Testing



Other Interesting Tools

- *Optic*: BC Break Detection
- *php-openapi-faker*: Create fake data from OpenAPI spec
- *Response2Schema*: Generate OpenAPI spec from JSON object
- *Laravel OpenAPI*: Generate OpenAPI spec from a Laravel app

Many more at https://openapi.tools


To sum up

Resources

- https://www.openapis.org
- https://openapi.tools

- https://github.com/thephpleague/openapi-psr7-validator
- https://github.com/akrabat/slim4-rps-api

Wandering Woodsman 🔥🚵🌳

@philsturgeon

If you've not got a test suite, YOU NEED A TEST SUITE.

If you've not got OpenAPI, why are you making every step of the API lifecycle worse, slower, and more manual.

API Design-First: https://apisyouwonthate.com/blog/api-design-first-vs-code-first
Or, play catchup: https://apisyouwonthate.com/blog/creating-openapi-from-http-traffic

Either way, go get OpenAPI.

11:49 AM · Feb 5, 2022 · Twitter Web App


Thank you!

Photo credits

- Scaffolding: https://www.flickr.com/photos/pagedooley/49683539647
- Writing: https://www.flickr.com/photos/throughkikslens/14516757158
- Books: https://www.flickr.com/photos/eternaletulf/41166888495
- Computer code: https://www.flickr.com/photos/n3wjack/3856456237
- Rocket launch: https://www.flickr.com/photos/gsfc/16495356966
- Stars: https://www.flickr.com/photos/gsfc/19125041621