

A sunset over the ocean with a dark silhouette of a rocky shore in the foreground. The sky transitions from a deep blue at the top to a bright orange near the horizon where the sun is setting. The water is dark with some white-capped waves in the foreground.

The How and Why of OpenAPI

Rob Allen

Web Summer Camp, July 2024

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

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

Metadata

```
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

paths:

 '/games':

 post:

 operationId: createGame

 summary: Create a new game

 description: Create a new game of Rock-Paper-Scissors.

 requestBody:

 # ...

 responses:

 # ...



Endpoints

paths:

'/games':

post:

operationId: createGame

summary: Create a new game

description: Create a new game of Rock-Paper-Scissors.

requestBody:

...

responses:

...



Endpoints

```
paths:
```

```
  '/games':
```

```
    post:
```

```
      operationId: createGame
```

```
      summary: Create a new game
```

```
      description: Create a new game of Rock-Paper-Scissors.
```

```
      requestBody:
```

```
        # ...
```

```
      responses:
```

```
        # ...
```



Endpoints

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



RequestBody

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

RequestBody

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



RequestBody

```
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

```
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

```
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

```
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"
```



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"}'



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"}'
```



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

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



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'
```



Responses

```
responses:
```

```
  '201':
```

```
    $ref: '#/components/responses/NewGameResponse'
```

```
  '400':
```

```
    $ref: '#/components/responses/NewGameError'
```

```
  '500':
```

```
    $ref: '#/components/responses/InternalServerError'
```



A close-up photograph of a person's hands using a purple highlighter on a document. The document features several examples of calligraphic text, including large, ornate letters and smaller, more fluid script. The scene is dimly lit, with the highlighter's tip and the person's fingers being the primary light sources. The text on the page is partially obscured by the highlighter and the hands.

Writing your spec



Editing

It's just text!

```
! rps-openapi.yaml x
doc > ! rps-openapi.yaml > [ ] tags > { } 0 > description
1  openapi: "3.0.3"
2
3  info:
4    title: Rock-Paper-Scissors
5    version: "1.0.0"
6    description: An implementation of Rock-Paper-Scissors
7    contact:
8      name: "Rob Allen"
9    license:
10     name: The MIT License
11
12  servers:
13    - url: https://rock-paper-scissors.example.com
14
15 > paths: ...
156
157 > components: ...
399
```

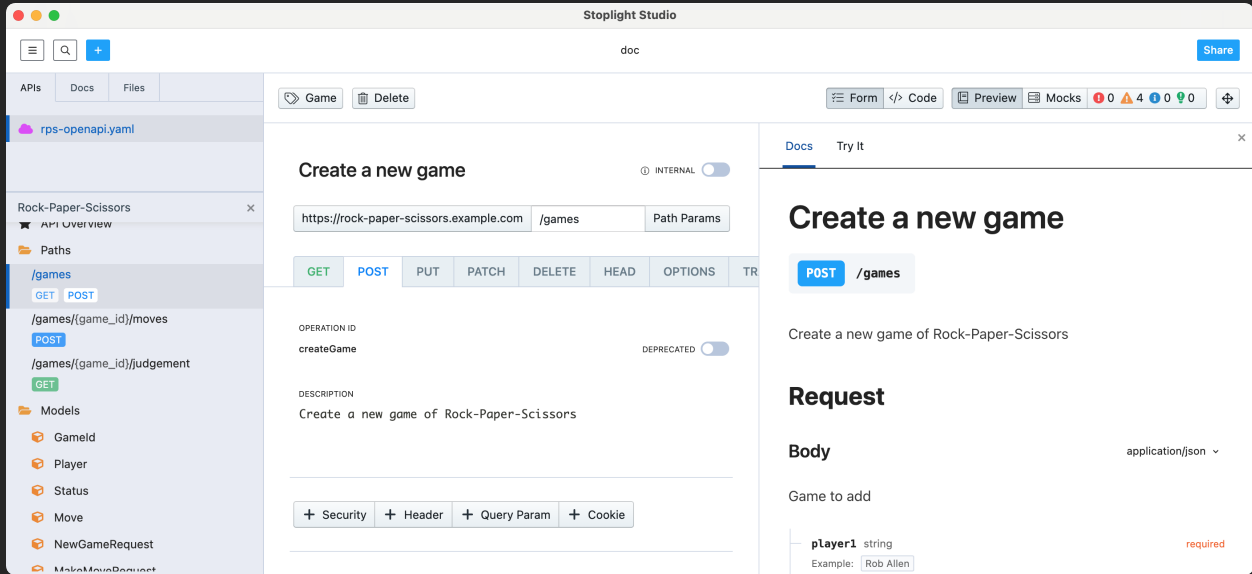
EXPLORER

- > OPEN EDITORS
- > SLIM4-RPS-API
- ▼ OUTLINE
 - openapi 3.0.3
 - ▼ { } info
 - title Rock-Paper-Scissors
 - version 1.0.0
 - description An implementation of ...
 - ▼ { } contact
 - name Rob Allen
 - ▼ { } license
 - name The MIT License
 - ▼ [] servers
 - ▼ { } 0
 - url https://rock-paper-scissors.ex...
 - ▼ { } paths
- > TIMELINE

Ln 404, Col 1 Spaces: 2 UTF-8 LF YAML No JSON Schema

Editing

GUI tools: Stoplight, OpenAPI-GUI, Swagger Editor



Linting & validation

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

```
$ spectral lint openapi.yaml
```

```
No results with a severity of 'error' or higher found!
```

Validation error

```
$ spectral lint openapi.yaml
```

```
.../slim4-rps-api/doc/openapi.yaml
```

```
3:6 warning info-contact Info object must have  
"contact" object. info
```

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



Coding Time!

Write an OpenAPI spec



Docs

Docs

The screenshot displays a REST client interface with a sidebar on the left containing a search bar and a list of API endpoints: GET List all games, POST Create a new game, POST Make a move, and GET Judge the game. The main area is titled 'Create a new game' and includes a description: 'Create a new game of Rock-Paper-Scissors'. Below this, the 'REQUEST BODY SCHEMA' is defined as 'application/json'. A table lists the required fields: 'player1' and 'player2', both of type 'string (Player)'. The 'Responses' section lists error codes: 201 (successful), 400 (invalid info), 404 (not found), 405 (method not allowed), and 500 (server error). On the right, a dark-themed panel shows the 'Request samples' section with a 'Payload' tab containing a JSON object: { "player1": "Lucy", "player2": "Dave" }. Below it, the 'Response samples' section shows a '201' status with a JSON response: { "game_id": "2BC08389-885A-4322-80D0-EF0DE2D7CD...", "_links": { "makeNextMove": { "href": "/games/2BC08389-885A-4322-80D0...", "description": "Make a player's move" } } }.

Create a new game

Create a new game of Rock-Paper-Scissors

REQUEST BODY SCHEMA: `application/json`

Field	Type
<code>player1</code> <small>required</small>	string (Player)
<code>player2</code> <small>required</small>	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

Request samples

POST /games

Content type: `application/json`

```
{
  "player1": "Lucy",
  "player2": "Dave"
}
```

Response samples

201

Content type: `application/hal+json`

```
{
  "game_id": "2BC08389-885A-4322-80D0-EF0DE2D7CD...",
  "_links": {
    "makeNextMove": {
      "href": "/games/2BC08389-885A-4322-80D0...",
      "description": "Make a player's move"
    }
  }
}
```

The screenshot shows the Plaid API documentation for the `/item/get` endpoint. The page is viewed in a browser window with the URL `plaid.com/docs/api/items/#itemget`. The left sidebar contains a navigation menu with categories like 'ALL DOCS', 'API', 'Overview', 'Libraries', 'API versioning', 'Postman Collection', 'Product endpoints', 'Item endpoints', 'Institution endpoints', 'Account endpoints and schemas', 'Token endpoints', 'Processor endpoints', 'Sandbox endpoints', and 'Webhooks'. The 'Item endpoints' section is expanded, showing `# /item/get`, `# /item/remove`, and `# /item/webhook/update`.

The main content area features a search bar, the API version '2020-09-14', and a 'Get API keys' button. The endpoint `/item/get` is titled 'Retrieve an Item' and is described as 'Returns information about the status of an Item.' Below this, there are tabs for 'Current libraries' and 'Legacy libraries'. The 'Request fields and example' section lists three fields: `client_id` (string), `secret` (string), and `access_token` (string, required). Each field has a description of its purpose. To the right, a code editor shows a Node.js example for making the request and handling the response.

The 'Response fields and example' section is partially visible, showing the `item` field (object) with the description 'Metadata about the Item.' To its right, another code editor shows the beginning of the API response object.

Docs

The screenshot shows the GitHub repository page for 'plaid / plaid-openapi'. The repository is in the 'master' branch, has 1.20.6+ versions, 19 branches, and 10 tags. The latest commit is by 'stephenjayakar' on 2020-09-14, 162.7 days ago, with 62 commits. The repository contains files like '.github/workflows', '2020-09-14.yml', 'CHANGELOG.md', and 'README.md'. The README.md file is open, showing the title 'plaid-OpenAPI' and a description: '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.' The section 'Using the OpenAPI generator' is also visible, with a link to 'You can find examples on the official OpenApiGenerator docs.' On the right side, there are statistics: API version 2020-09-14, 39 stars, 35 watching, 18 forks, and 1 year old. There is also a 'Contributors' section with 8 contributors.

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

stephenjayakar 2020-09-14_162.7 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

The screenshot shows a GitHub repository page for `Nexmo / api-specification`. The repository has 12 issues, 8 pull requests, 5 actions, and 193 releases. The current file being viewed is `application.v2.yml` in the `definitions` directory. The file was last committed by `francesco-fipertani-vonage` on 9 Nov 2021. The file content is as follows:

```
1 ---
2 openapi: "3.0.0"
3 info:
4   version: 2.1.1
5   title: "Application API"
6   description: |
7     Vonage provides an Application API to allow management of your Vonage Applications.
8
9   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
10  contact:
11    name: Vonage
12    url: "https://developer.nexmo.com/"
13    email: devrel@nexmo.com
14  servers:
15    url: https://api.nexmo.com/v2/applications
16  security:
17    basicAuth: []
18  paths:
19    /:
```


The screenshot shows the Vonage Developer API documentation page for the Application API. The page is titled "Application API" and includes a navigation bar with links for "Use Cases", "Documentation", "SDKs & Tools", and "Community". The main content area is divided into sections: "Available Operations" (with a sub-section for "Errors"), "List available applications", "Authentication", and "Query Parameter". The "Available Operations" section lists five methods: GET (List available applications), POST (Create an application), GET (Get an application), PUT (Update an application), and DELETE (Delete an application). The "List available applications" section shows a GET request to "https://api.nexmo.com/v2/applications". The "Authentication" section includes a table with columns for Key, Description, Example, and Default. The "Query Parameter" section is partially visible. On the right side, there is a warning box stating "There are multiple versions of this API available" with links for "VERSION 1" and "VERSION 2". Below this, there is an "Example Responses" section with a dropdown menu showing status codes 200, 400, 401, 405, and 406. The 200 response is selected, showing a JSON object with fields like "page_size", "page", "total_items", "total_pages", and "applications". A "Feedback" button is located on the right edge of the page.

developer.vonage.com/api/application.v2

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

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

There are multiple versions of this API available
VERSION 1 | VERSION 2

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",

```

Demo Time!

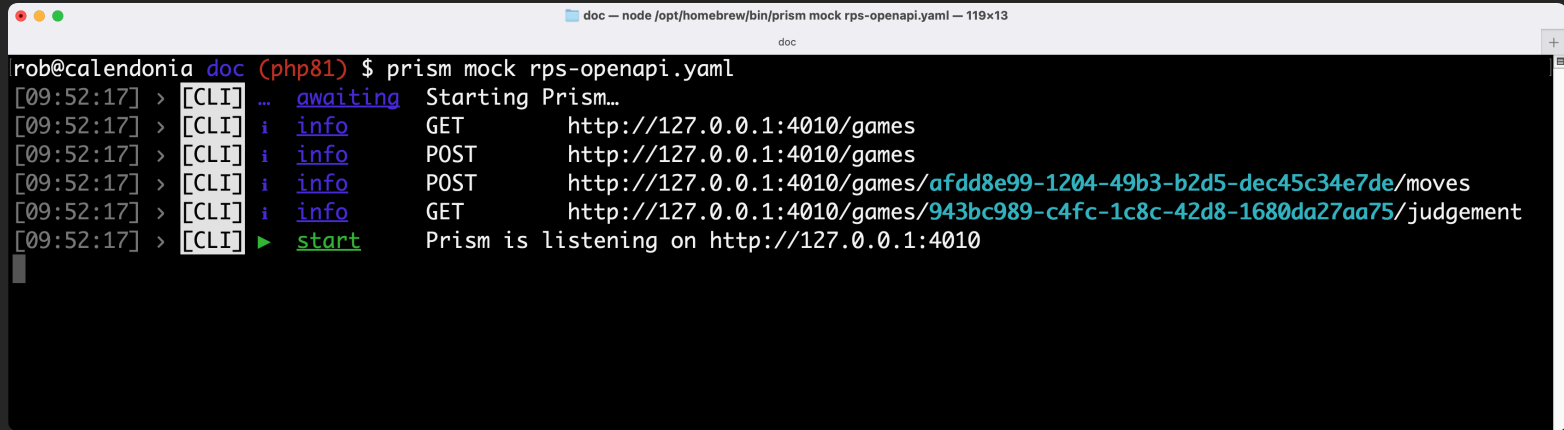
Generating docs



Developers

Mock server

```
$ prism mock openapi.yaml
```

A terminal window with a dark background and light text. The window title is "doc - node /opt/homebrew/bin/prism mock rps-openapi.yaml - 119x13". The prompt is "rob@calendonia doc (php81) \$". The command "prism mock rps-openapi.yaml" has been executed. The output shows Prism starting and listening on http://127.0.0.1:4010. It lists five mock endpoints: GET http://127.0.0.1:4010/games, POST http://127.0.0.1:4010/games, POST http://127.0.0.1:4010/games/afdd8e99-1204-49b3-b2d5-dec45c34e7de/moves, GET http://127.0.0.1:4010/games/943bc989-c4fc-1c8c-42d8-1680da27aa75/judgement, and a green arrow pointing to "start".

```
rob@calendonia doc (php81) $ prism mock rps-openapi.yaml
[09:52:17] > [CLI] ... awaiting Starting Prism...
[09:52:17] > [CLI] i info GET http://127.0.0.1:4010/games
[09:52:17] > [CLI] i info POST http://127.0.0.1:4010/games
[09:52:17] > [CLI] i info POST http://127.0.0.1:4010/games/afdd8e99-1204-49b3-b2d5-dec45c34e7de/moves
[09:52:17] > [CLI] i info GET http://127.0.0.1:4010/games/943bc989-c4fc-1c8c-42d8-1680da27aa75/judgement
[09:52:17] > [CLI] ► start Prism is listening on http://127.0.0.1:4010
```

Make API calls

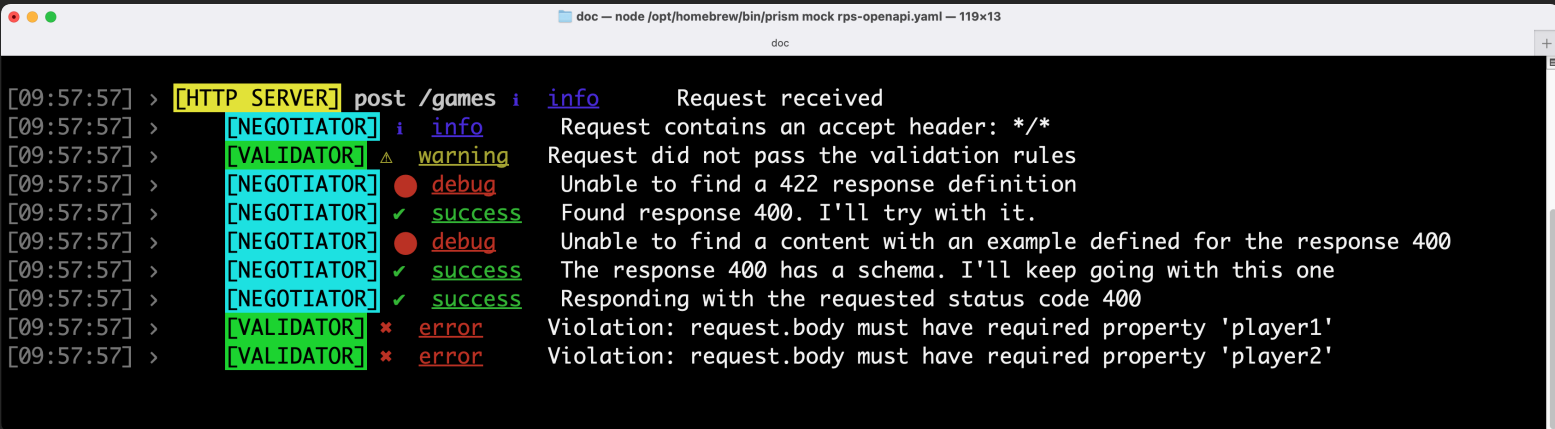
```
$ curl http://127.0.0.1:4010/games -d '{}'
```

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"}
```



The screenshot shows a terminal window with the following content:

```
doc -- node /opt/homebrew/bin/prism mock rps-openapi.yaml -- 119x13  
doc  
[09:57:57] > [HTTP SERVER] post /games i info Request received  
[09:57:57] > [NEGOTIATOR] i info Request contains an accept header: /*/  
[09:57:57] > [VALIDATOR] Δ warning Request did not pass the validation rules  
[09:57:57] > [NEGOTIATOR] ● debug Unable to find a 422 response definition  
[09:57:57] > [NEGOTIATOR] ✓ success Found response 400. I'll try with it.  
[09:57:57] > [NEGOTIATOR] ● debug Unable to find a content with an example defined for the response 400  
[09:57:57] > [NEGOTIATOR] ✓ success The response 400 has a schema. I'll keep going with this one  
[09:57:57] > [NEGOTIATOR] ✓ success Responding with the requested status code 400  
[09:57:57] > [VALIDATOR] * error Violation: request.body must have required property 'player1'  
[09:57:57] > [VALIDATOR] * error Violation: request.body must have required property 'player2'
```



Demo Time!

Using a mock server



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!

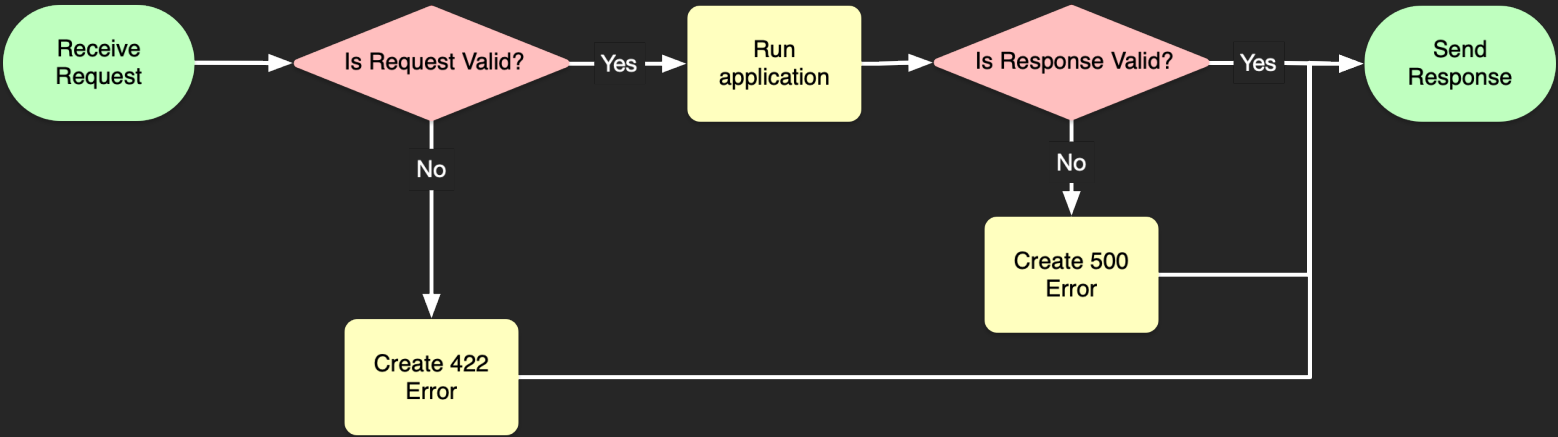


Validation in PHP

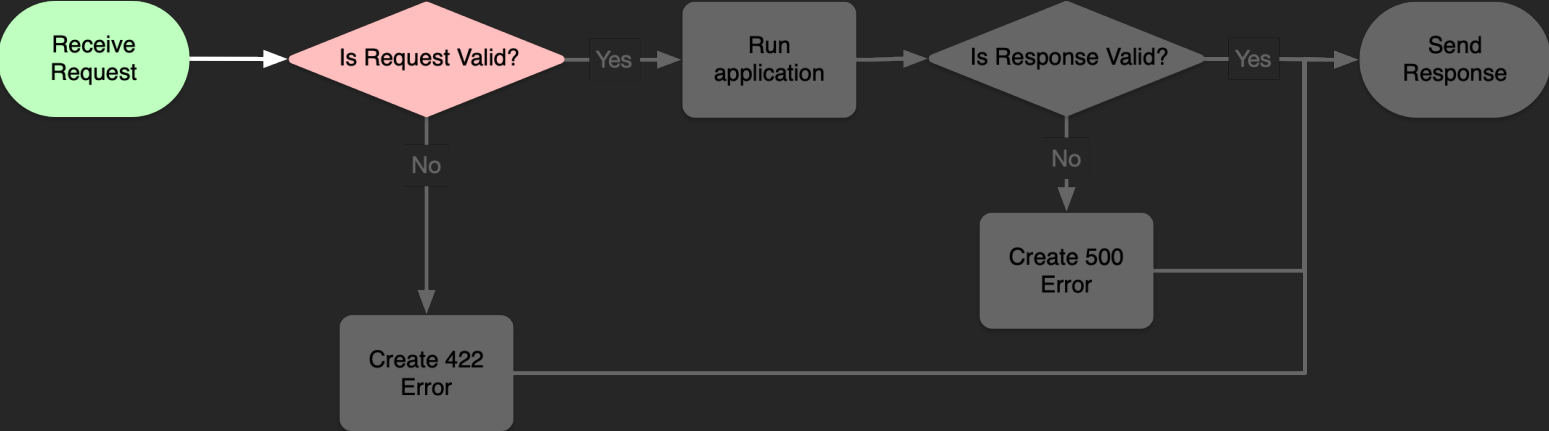
- `league/openapi-psr7-validator`
- `opis/json-schema`



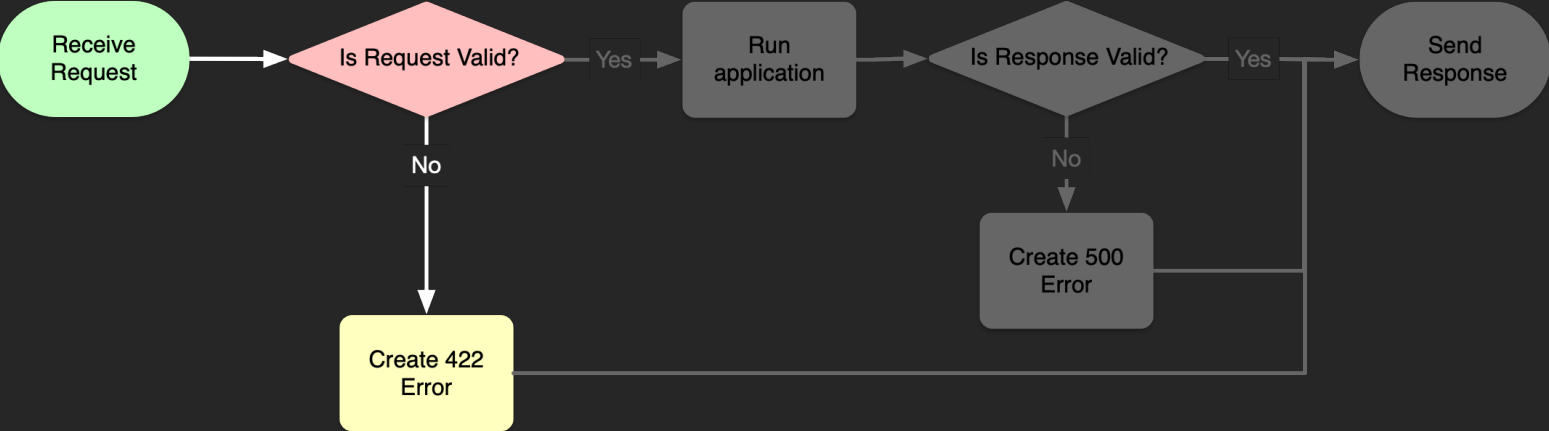
Validation middleware



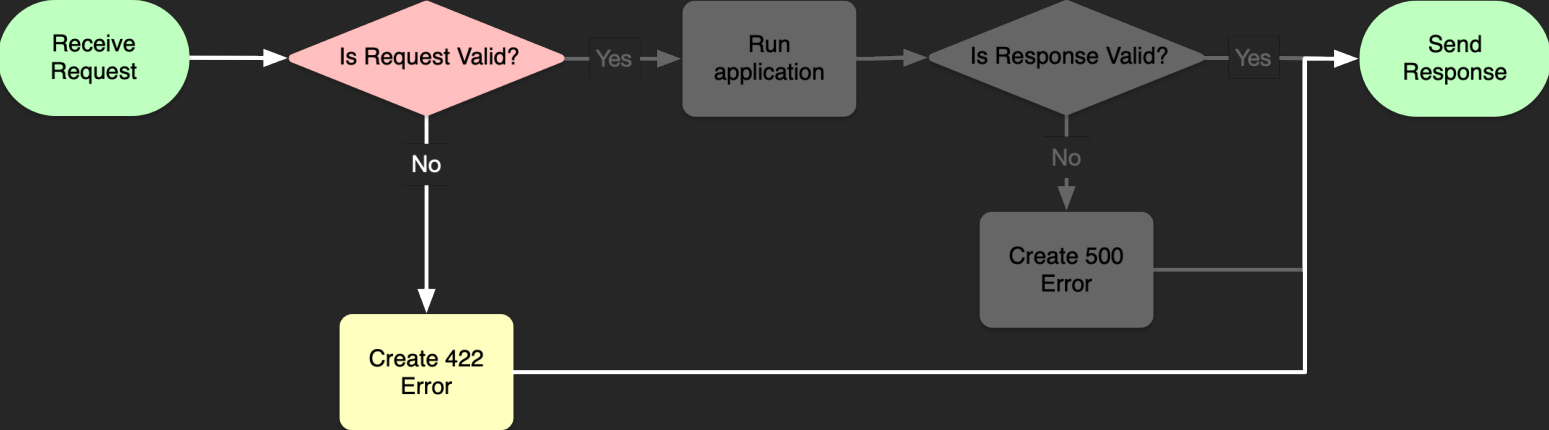
Test Request



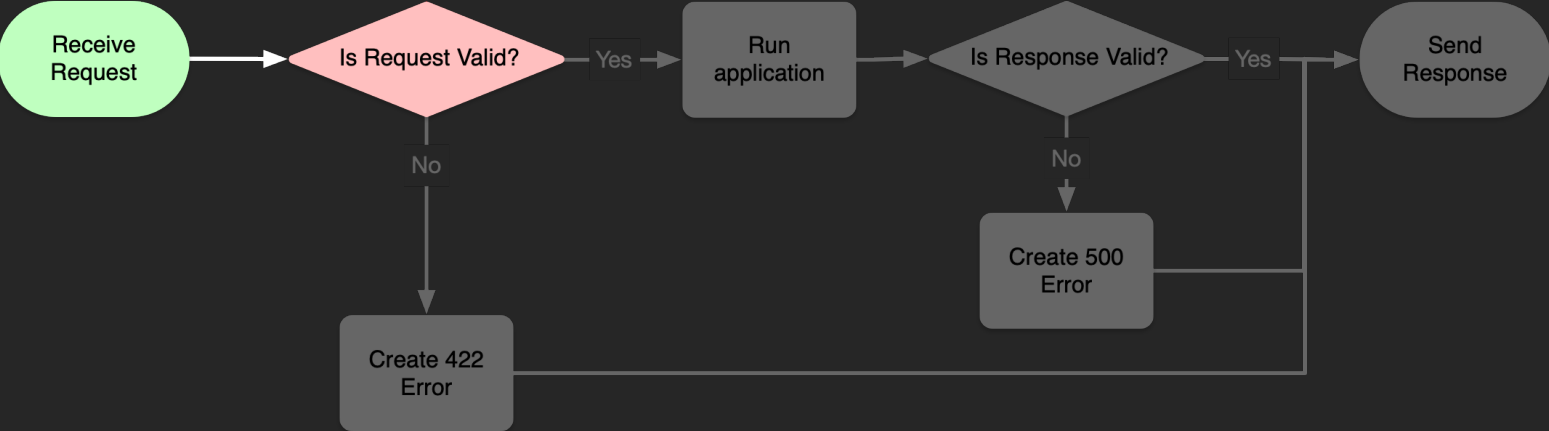
Request is invalid



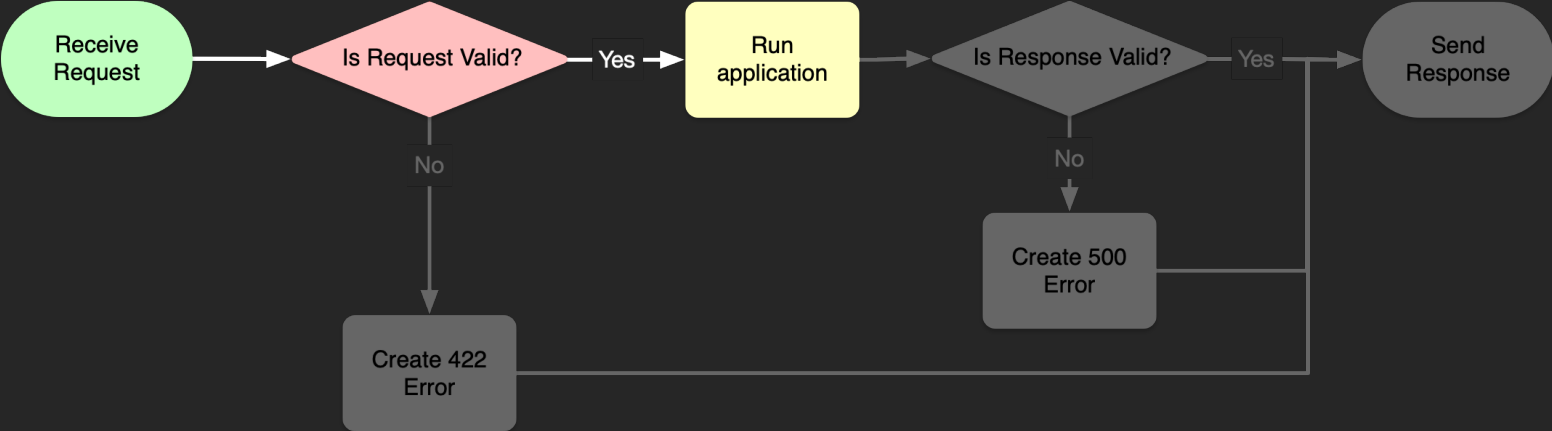
Request is invalid



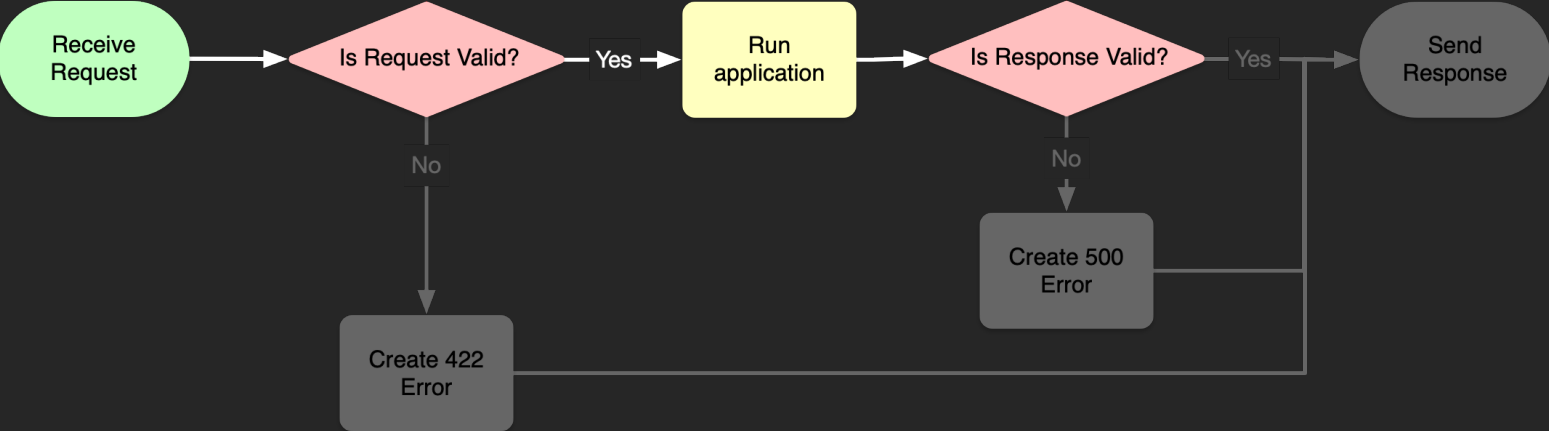
Test Request



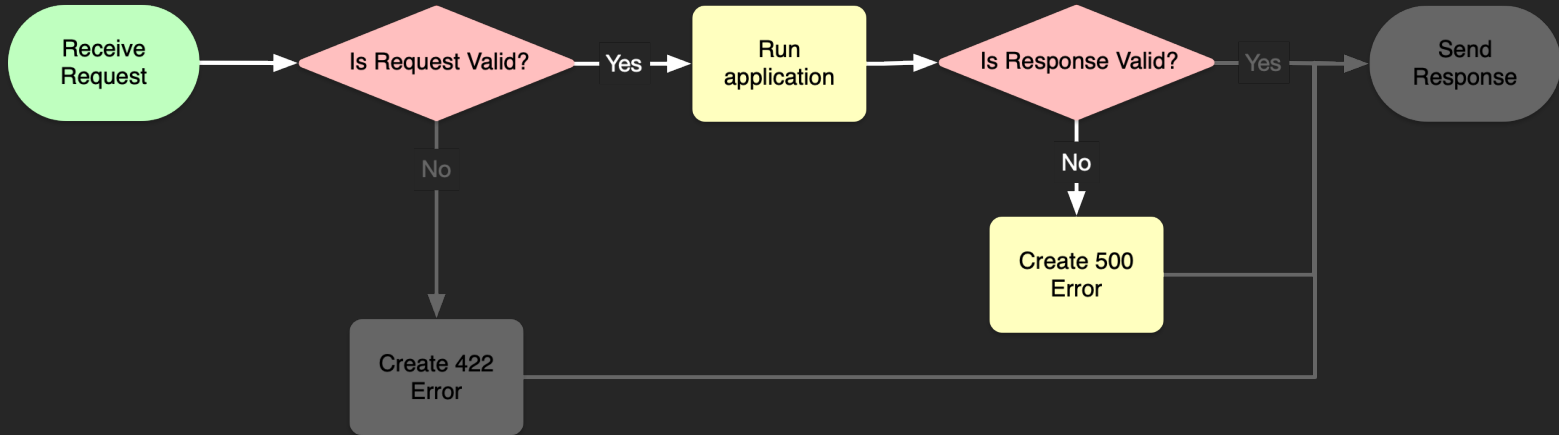
Request is valid



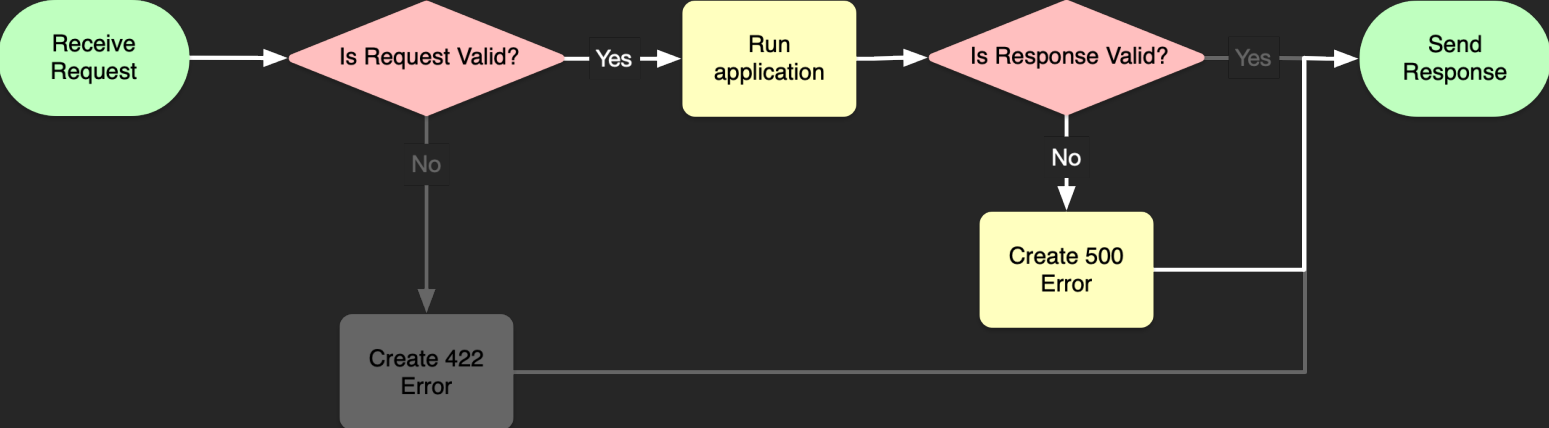
Test Response



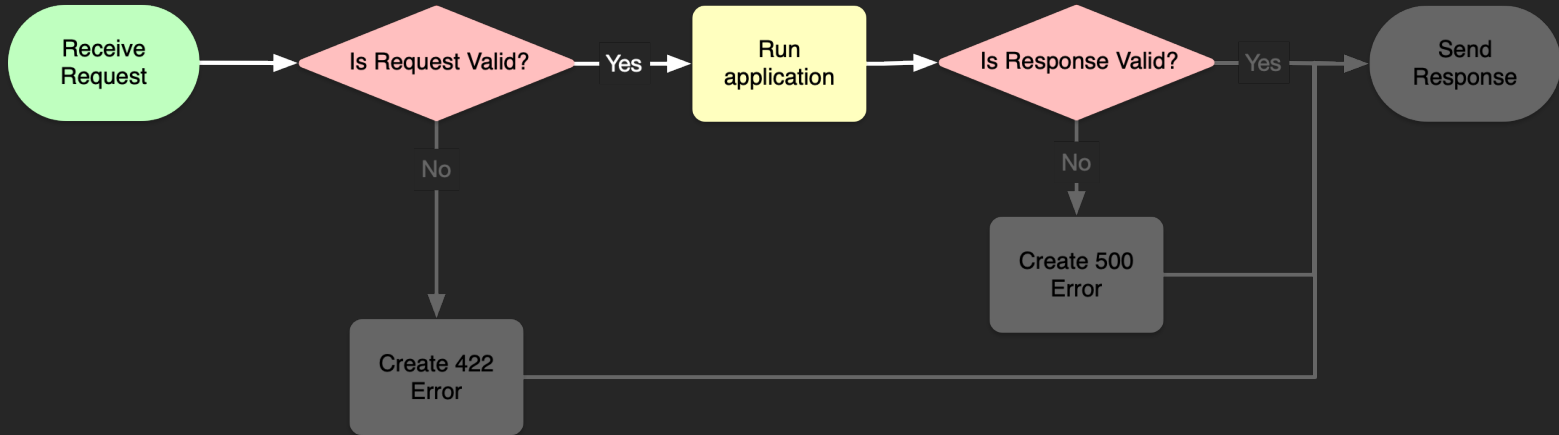
Response is invalid



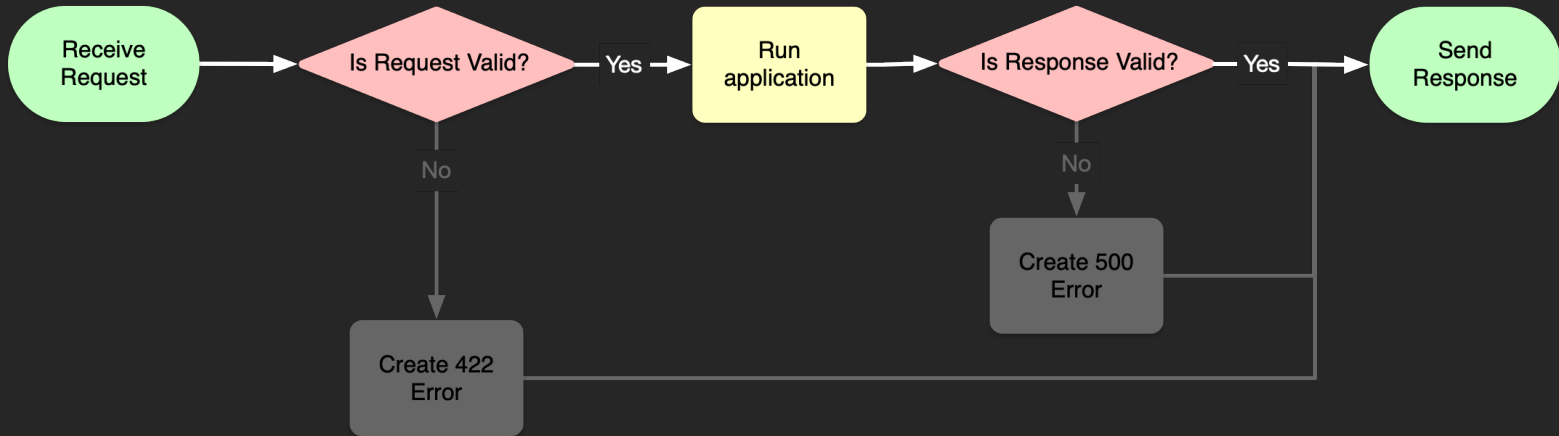
Response is invalid



Successful validation



Successful validation



Validation middleware

```
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);
    }
}
```

Coding Time!

Validating a PHP API



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/openapi.yaml
```

Compliance Testing

```
doc -- doc -- -bash -- 119x26
===== Schemathesis test session starts =====
platform Darwin -- Python 3.9.7, schemathesis-3.12.3, hypothesis-6.36.0, hypothesis_jsonschema-0.22.0, jsonschema-4.4.0
rootdir: /Users/rob/Projects/slimng/slim4-rps-api/doc
hypothesis profile 'default' -> database=DirectoryBasedExampleDatabase('/Users/rob/Projects/slimng/slim4-rps-api/doc/.hypothesis/examples')
Schema location: file:///Users/rob/Projects/slimng/slim4-rps-api/doc/rps-openapi.yaml
Base URL: http://localhost:8888
Specification version: Open API 3.0.3
Workers: 1
Collected API operations: 4

GET /games . [ 25%]
POST /games . [ 50%]
POST /games/{game_id}/moves . [ 75%]
GET /games/{game_id}/judgement . [100%]

===== SUMMARY =====

Performed checks:
not_a_server_error          306 / 306 passed PASSED
status_code_conformance    306 / 306 passed PASSED
content_type_conformance   306 / 306 passed PASSED
response_headers_conformance 306 / 306 passed PASSED
response_schema_conformance 306 / 306 passed PASSED

===== 4 passed in 85.55s =====
```



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>

A photograph of a rocket launch at dusk or dawn. The rocket is ascending vertically, leaving a large, bright plume of white and orange smoke and fire. Several tall, thin service towers are visible around the launch pad. The sky is a deep orange and red, and the ground is dark. The text "To sum up" is overlaid in a yellow, serif font in the upper center of the image.

To sum up

Resources

- <https://www.openapis.org>
- <https://openapi.tools>
- <https://github.com/the-php-league/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/gsfcr/16495356966>
- Stars: <https://www.flickr.com/photos/gsfcr/19125041621>