Reference links

<https://github.com/manideep-vv/forked-openapi/tree/main/openapi-yaml-examples>

sample open api examples

<https://explore.swaggerhub.com/> same like postman with this we can retrieve the swagger documentation

<https://swapi.dev/> (sw – star wars api) 🡪 Dummy rest api to get some data

<https://reqres.in/> 🡪 this is also dummy rest api

openAPI map <https://openapi-map.apihandyman.io/> helpful website to know openAPI structure <https://openapi-map.apihandyman.io/?version=3.0>

Problems before open api

1. No proper documentation, for end points info we need to ask developers , so if they gave old url that’s it
2. There is no standard of developing api

What is open API?

Open API specifies the way of describing HTTP REST API’s

An open API definition comes in the form of YAML or JSON file which describes the input and output of API’s, that json is understandable by swagger ui tool

Open API spec == is nothing but a json or yaml file

2010 swagger born, 2015 swagger donated the specification part to Linux foundation & they renamed the spec part to open API & smart bear retained the copyright for swagger

So specification is called OPEN API and products are called swagger

Advantages

1. With that spec (json/yaml file) we can generate both RESTful server side and client side code (producer and consumer codes) with help of swagger codeGen tool
2. We can perform the validations on data through ur api (means we don’t need to write the code to validate the data in producer side )

Adv of swagger

We can give our openAPI spec to the consumers who want to consume

1. They can see all endpoint info, like http method type (put/post/get …) and parameters info like number of params, type of params ex:- employee type..
2. Each env url – dev,prod,qa …
3. people can generate the stubs I mean producer side code and consumer side code to consume the api or to create REST controller code

Swagger tools

1. Swagger code gen – free- this is to generate the stubs producer and consumer side java / any language code
2. Swagger editor –free- with this editor we can create and edit open API spec, API editor for designing api’s with open api and async api specifications

<https://editor.swagger.io/> same like with Intellij we can create java code, with this editor we can create spec

if u are able to edit the spec means it is swagger editor

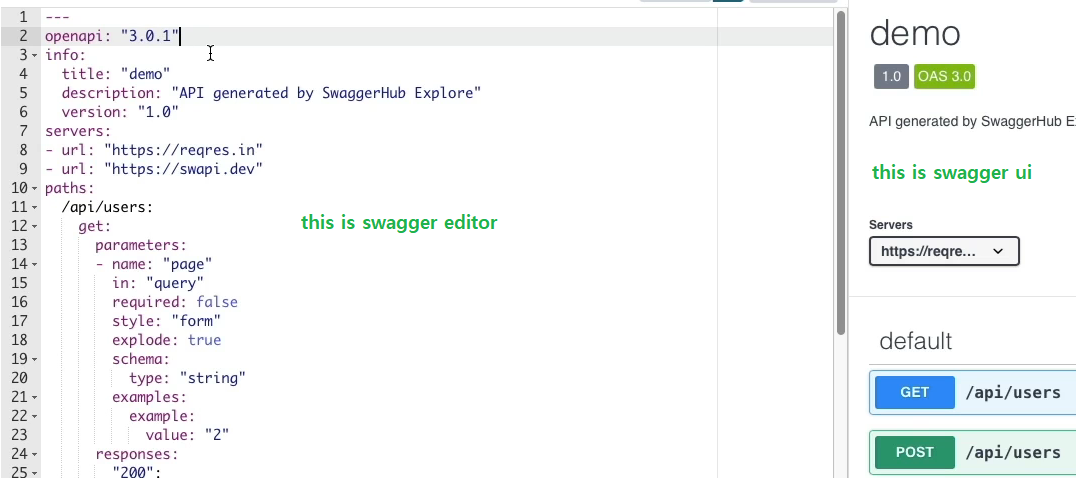
1. Swagger ui – to visualise the open api spec in an interactive ui – in below image
2. Swagger hub is a paid tool – like a git hub repo used to store the open api spec for our rest projects (here we can define our API’s and manage them like github)
3. Swagger hub explorer –(this is like a postman tool ) using swagger hub explore 1) we can test already available API’s 2) **we can generate open api spec**

<https://explore.swaggerhub.com/> same like postman with this we can retrieve the swagger documentation

Sample swagger ui

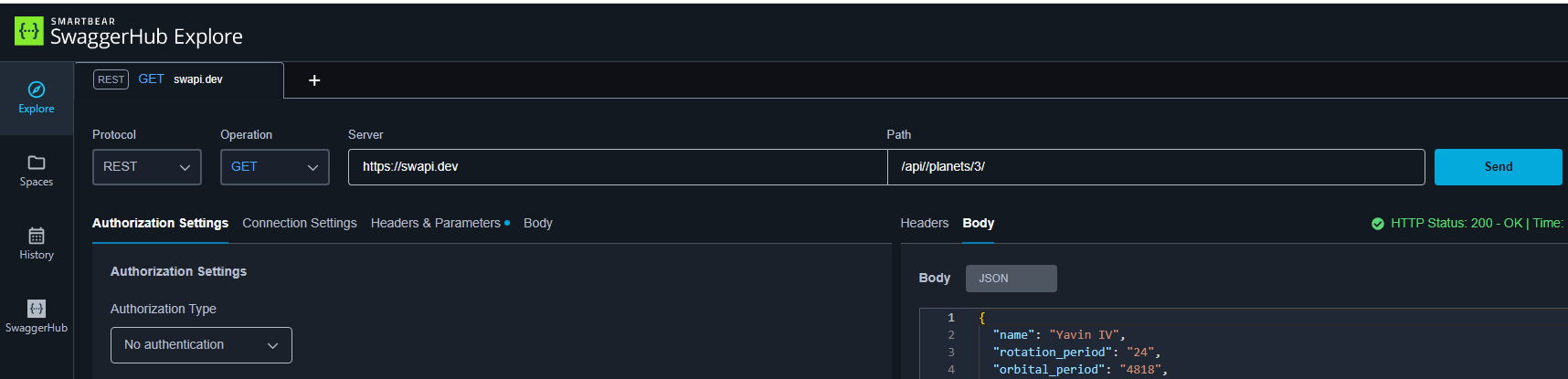
It will take our open api spec and generate display our endpoints

Here left side is swagger editor, right side is swagger ui

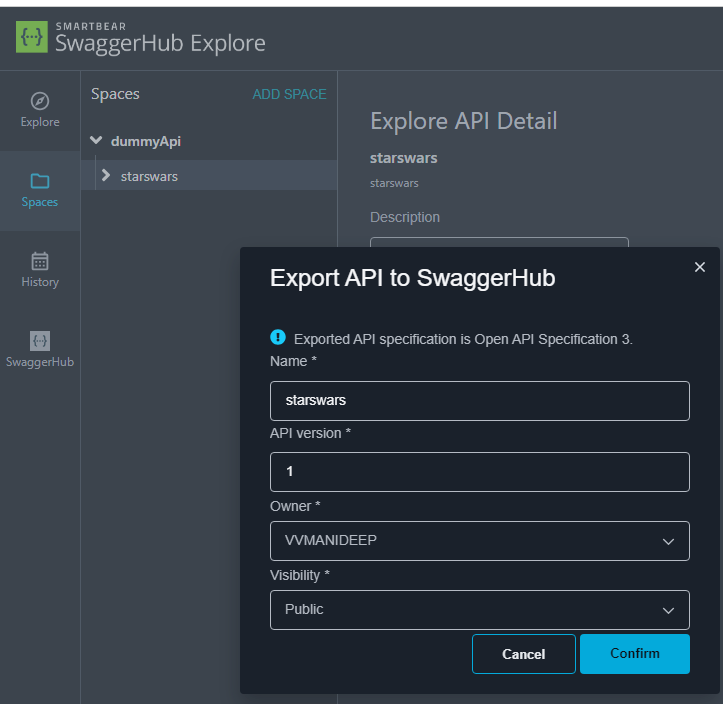


Generate open API spec from endpoint

1. Get url of url rest api -- <https://swapi.dev/>
2. Goto swagger hub explorer <https://explore.swaggerhub.com/> go to explore tab and hit API as this is like a postman we can hit api & test



1. Go to history and select that api or list of api’s & click “Add to space” after that
2. Go to spaces 🡪 select any api and then we have to export that spec to swagger hub as below
3. Then u will navigate to swagger hub & u can see the spec as per above image



1. Fg

Design first approach

In design first approach first we will write openAPI spec using swagger editor tool, and it’s possible to generate the stubs (REST producer or consumer side code)

from that openapi spec

Where as in code first approach, 1st we will code and we will generate open API spec using that endpoint, we can give that spec whom ever it is needed

The advantage is if we already have spec, with codeGen tools we can generate the server side client side code like rest controller code automatically

Common mark / open mark syntax

In git hub also we will use this common mark syntax only in github readme.md file

# single for big headings h1

## means h2

### means h3 for even smaller

|  |  |
| --- | --- |
| \* broo \* single star | For italic |
| \*\* broo \*\* double star | For bold |
| \*\*\* broo \*\*\* triple star | For bold and italic |
| \_API\_ surrounded by underscore | For italic |
| `orayya` back tick | For violet colour |
| # categories supported  - mobiles  - camera  - sony  - pony  - laptops  - monitors |  |

Tagging API’s

We should tag api to neatly organise, so that certain apis comes under certain category

Data types

integer – for positive and negative numbers

number- for normal and floating point number

string – use this even for date data types

ex:-files are also defined as strings

ex:- format:- binary (for binary file content) or byte (base 64 encoded content)

Boolean – true or false,

nullable :true so that it will allows null by default all are mandatory fields, we should give this to make that field as optional

array,

object:-

by default all ur object properties are optional, if u want to mention them as mandatory use required flag

schema:

type: object

properties:

productId:

type: integer

name:

type: string

price:

type: integer

required :

-productId

-name