OpenAPI Specification

RESTful web services documentation

University of Fribourg
Department of Informatics
Software Engineering Group

Outline

1. Introduction
2. Tools
3. Specification
4. References

Swagger and OpenAPI

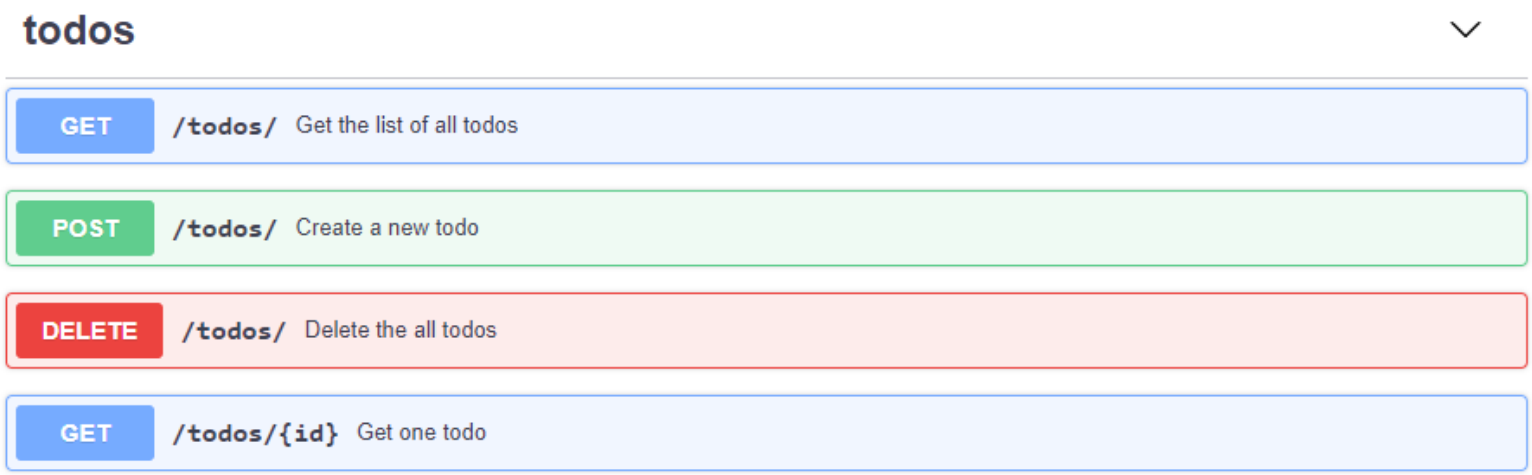
- Swagger is an open-source set of tools to help design RESTful web services, developed by SmartBear from 2011.
- A specification for describing REST APIs was developed within this framework.
- In 2016, this specification was renamed OpenAPI and moved to a standalone project.
- For this reason, what we generally call “Swagger” is in fact the OpenAPI specification.

Use Cases

- We often see OpenAPI documents used as a reference to understand an existing API.
- However, the strength of OpenAPI is that it can be used as a specification for developers who will create an API.
  - This document is often created before any implementation takes place.

Swagger UI

- Swagger UI is a tool that reads an OpenAPI document and generates a web client.
- This client is used to interact with the documented web service, which is helpful for early testing.



Generating a Document

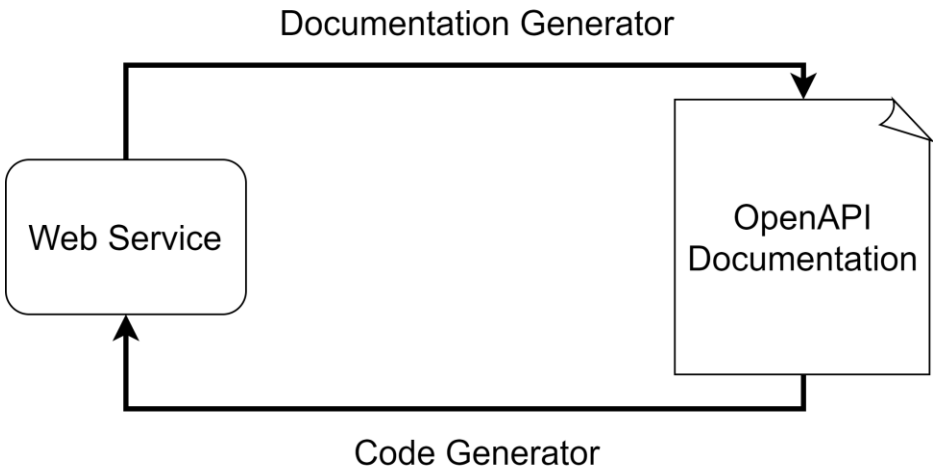
Swagger Editor

- Swagger Editor (https://editor.swagger.io/) is an online tool for creating and editing OpenAPI documents.
- The editor instantly generates a web client to directly interact with an existing web service.
- Yaml can used instead of JSON, the two languages are totally interchangeable.

Generating a Document

Automated Tools

- Documentation generators create a document from an existing web service.
- Using an existing document, code generators can create an application skeleton.
- A list of existing implementations of these tools is available online.



Top-Level Fields (1/3)

- An OpenAPI specification has a clearly-defined structure.
- These slides are just an introduction to the specification, see https://swagger.io/specification/ for the full documentation.
- At the root of the document, the following fields exist:
  - openapi: the OpenAPI version.
  - info: a description of the API.
  - servers (optional): an array of servers for the spec to interact with.
  - paths: an object describing the different routes of the API.

Top-Level Fields (2/3)

- components (optional): an object for describing resource schemas. These can then be referenced in the path objects.
- security (optional): an object describing the different security schemes used by the API.
- tags (optional): an array of tags used to group operations together (e.g. “todos” and “tags”).
- externalDocs (optional): an object describing additional documentation.

Top-Level Fields (3/3)

Specification Example


```yaml
openapi: 3.0.1
info:
  title: Todo Backend
  version: '1.0'
servers:
  - url: 'http://localhost:8080'
  - url: 'http://todo.thing.zone'
tags:
  - name: todos
  - name: tags
paths:
  /tags/:
    get:
      tags:
        - tags
      summary: Get the list of all tags
      responses:
        '200':
          description: List of all tags
          content:
            application/json:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/Tag'
components:
  schemas:
    Tag:
      description: Object representing a Tag
      type: object
      properties:
        id:
          description: id of the tag
          type: string
          example: '42'
        title:
          description: title of the tag
          type: string
          example: Leisure
        url:
          description: url associated with the tag
          type: string
```

Paths

- A path regroups the different path items that correspond to a given endpoint (e.g. “/todos({todoId}”).
- Path items can be:
  - method names (e.g. “get” or “post”) describing each one operation on the path.
  - summary.
  - description.
  - parameters: an array of the parameters used in the path (e.g. “todoId”).
    - Parameters can be defined at the path level or at the operation level.

Components

- Components are reusable pieces of information that can be referenced by other objects in the document:
  - Schemas are generally used to describe request bodies and responses.
  - Other components include responses, parameters, requestBodies, headers, …

References II

- Swagger
  https://swagger.io/
- The OpenAPI Specification.
  https://swagger.io/specification/