SWAGGER

REST API Documentation

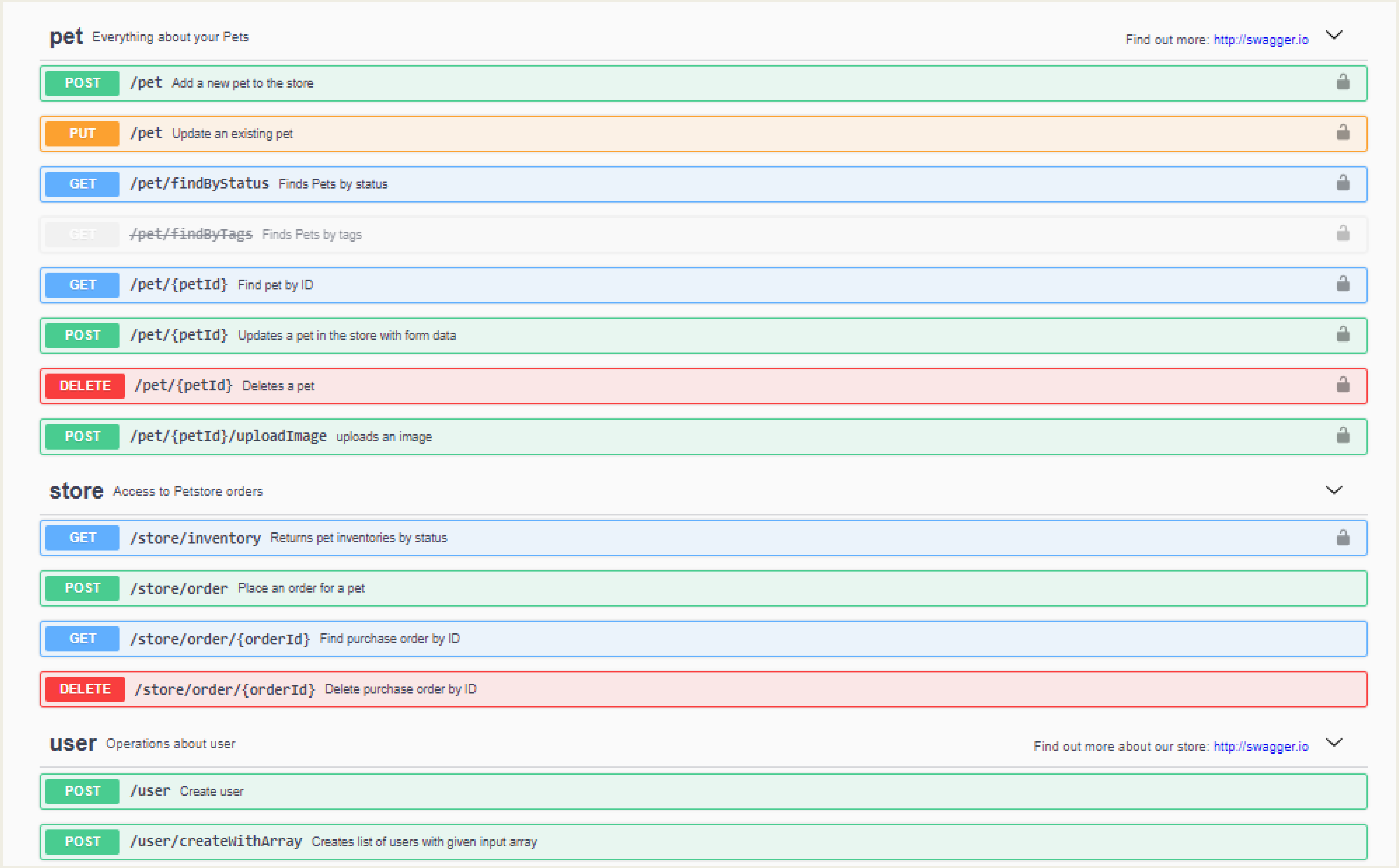
The Problem

- SOAP Web Services are exposed to other client applications with the help of WSDL.
- When we develop a REST API Based service, there is no such document available to expose the API to public and discover the API.
- The Absence of such API Documentation makes testing of REST and communicating it to the client very difficult.
  - *What URL, which method, how many parameters etc etc.*
- Hence a standard spec for REST API Documentation is required to expose your REST Service which also would provide discoverabillty.
- This is where **SWAGGER** Comes into the picture.

WHAT IS SWAGGER?

- The goal of Swagger™ is to define a standard, language-agnostic interface to REST APIs which allows both humans and computers to discover and understand the capabilities of the service without access to source code, documentation, or through network traffic inspection.
- Swagger is a **formal specification** surrounded by a large ecosystem of tools
- Swagger is now called **OpenAPI** Specification.

REST API Doc Using Swagger



pet Everything about your Pets — Find out more: http://swagger.io

POST /pet Add a new pet to the store

PUT /pet Update an existing pet

GET /pet/findByStatus Finds Pets by status

GET ~~/pet/findByTags~~ Finds Pets by tags

GET /pet/{petId} Find pet by ID

POST /pet/{petId} Updates a pet in the store with form data

DELETE /pet/{petId} Deletes a pet

POST /pet/{petId}/uploadImage uploads an image

store Access to Petstore orders

GET /store/inventory Returns pet inventories by status

POST /store/order Place an order for a pet

GET /store/order/{orderId} Find purchase order by ID

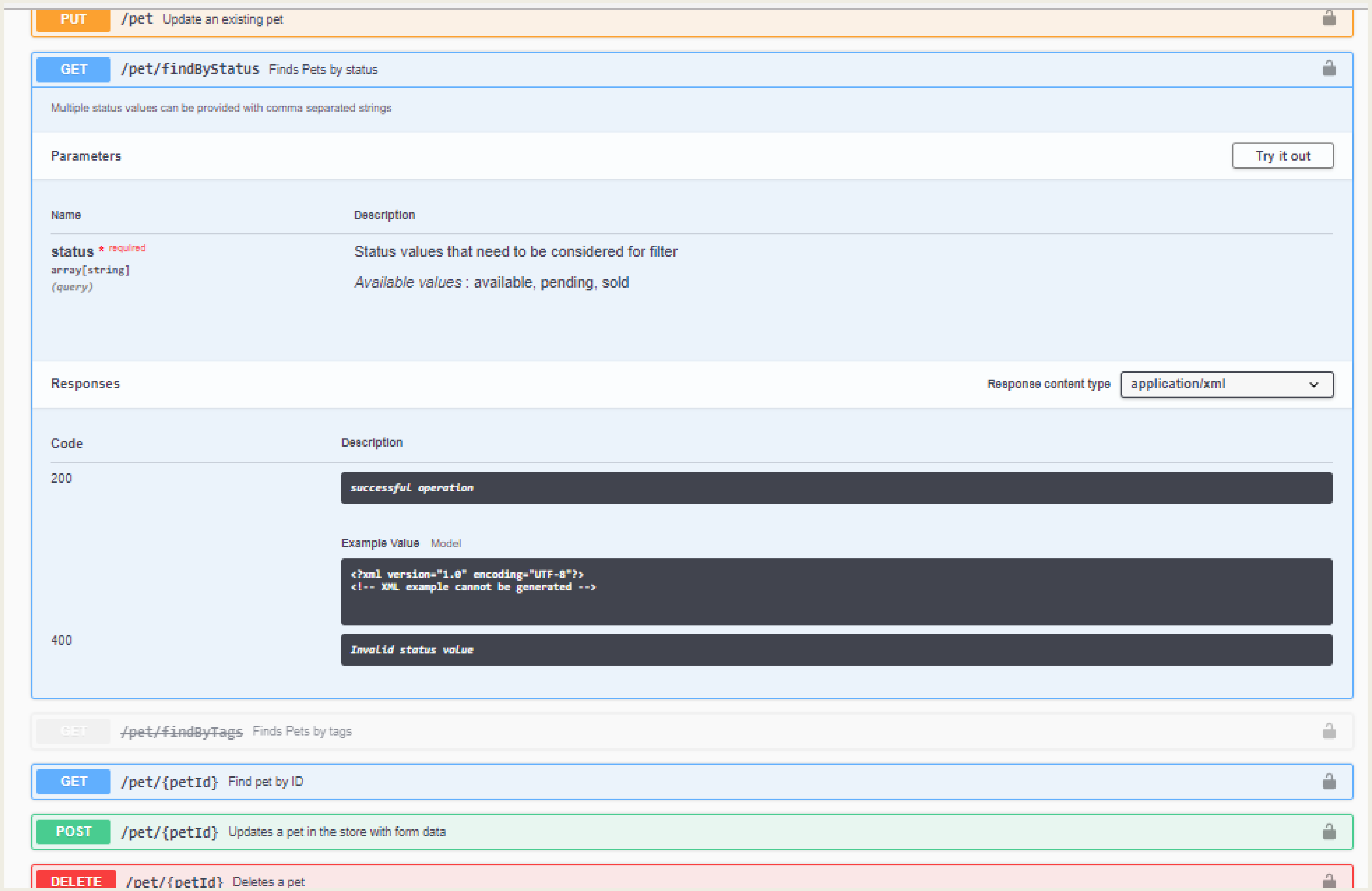
DELETE /store/order/{orderId} Delete purchase order by ID

user Operations about user — Find out more about our store: http://swagger.io

POST /user Create user

POST /user/createWithArray Creates list of users with given input array

REST API Doc Using Swagger



PUT /pet Update an existing pet

GET /pet/findByStatus Finds Pets by status

Multiple status values can be provided with comma separated strings

Parameters

Name	Description
status * required array[string] (query)	Status values that need to be considered for filter *Available values* : available, pending, sold

Responses

Response content type application/xml

Code	Description
200	successful operation
400	Invalid status value

Example Value Model

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- XML example cannot be generated -->
```

GET ~~/pet/findByTags~~ Finds Pets by tags

GET /pet/{petId} Find pet by ID

POST /pet/{petId} Updates a pet in the store with form data

DELETE /pet/{petId} Deletes a pet

SWAGGER ECOSYSTEM

- **Swagger Editor**
  - *edit API specifications in YAML inside browser and preview documentations in real time.*
- **Swagger Codegen**
  - *allows generation of both client libraries and server stubs from a Swagger definition.*
- **Swagger UI**
  - *dependency-free collection of HTML, Javascript, and CSS assets that dynamically generate beautiful documentation from a Swaggercompliant*
- API

http://swagger.io/tools/

USAGE PATTERNS FOR API PROVIDERS

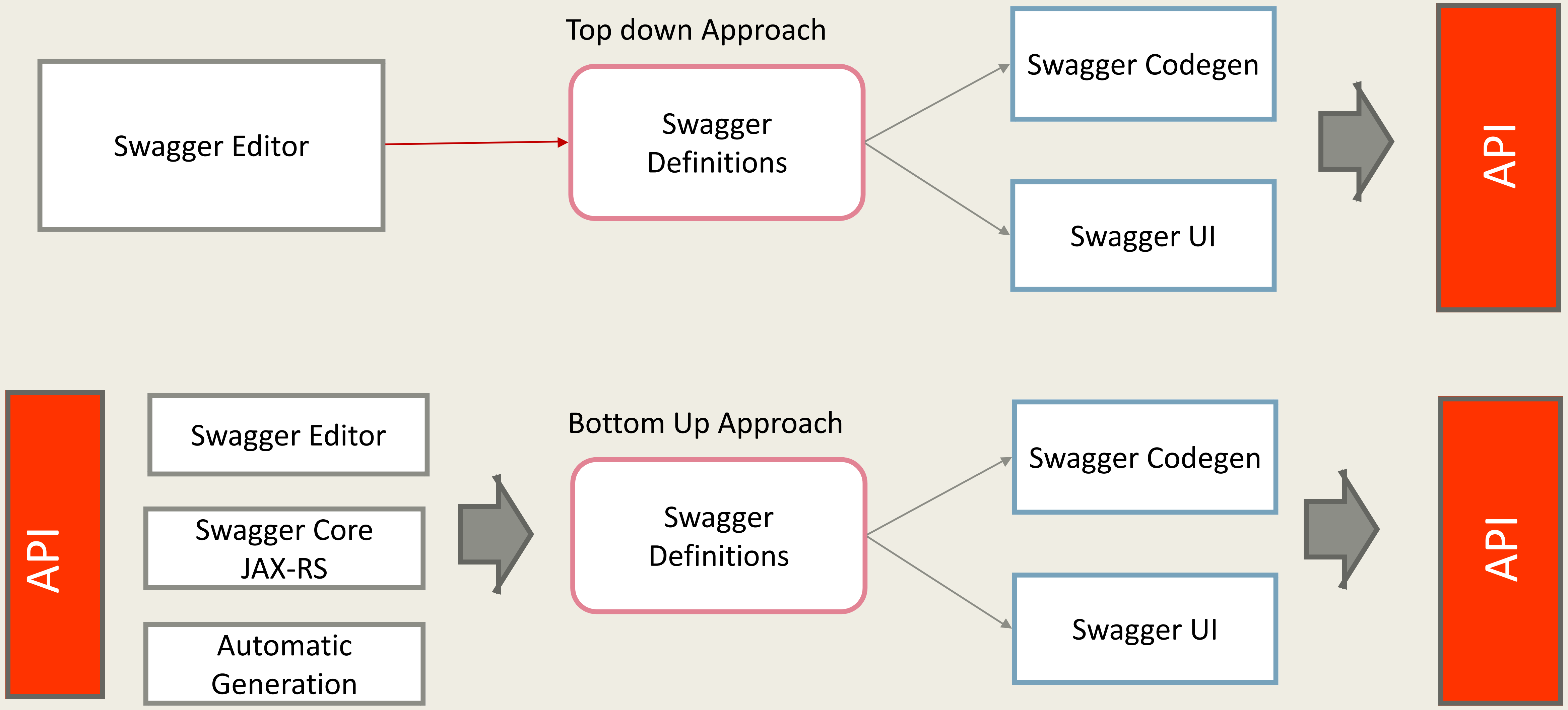


Image concept from Andrii Gakhov techtalk@ferret

SWAGGER SPECIFICATION

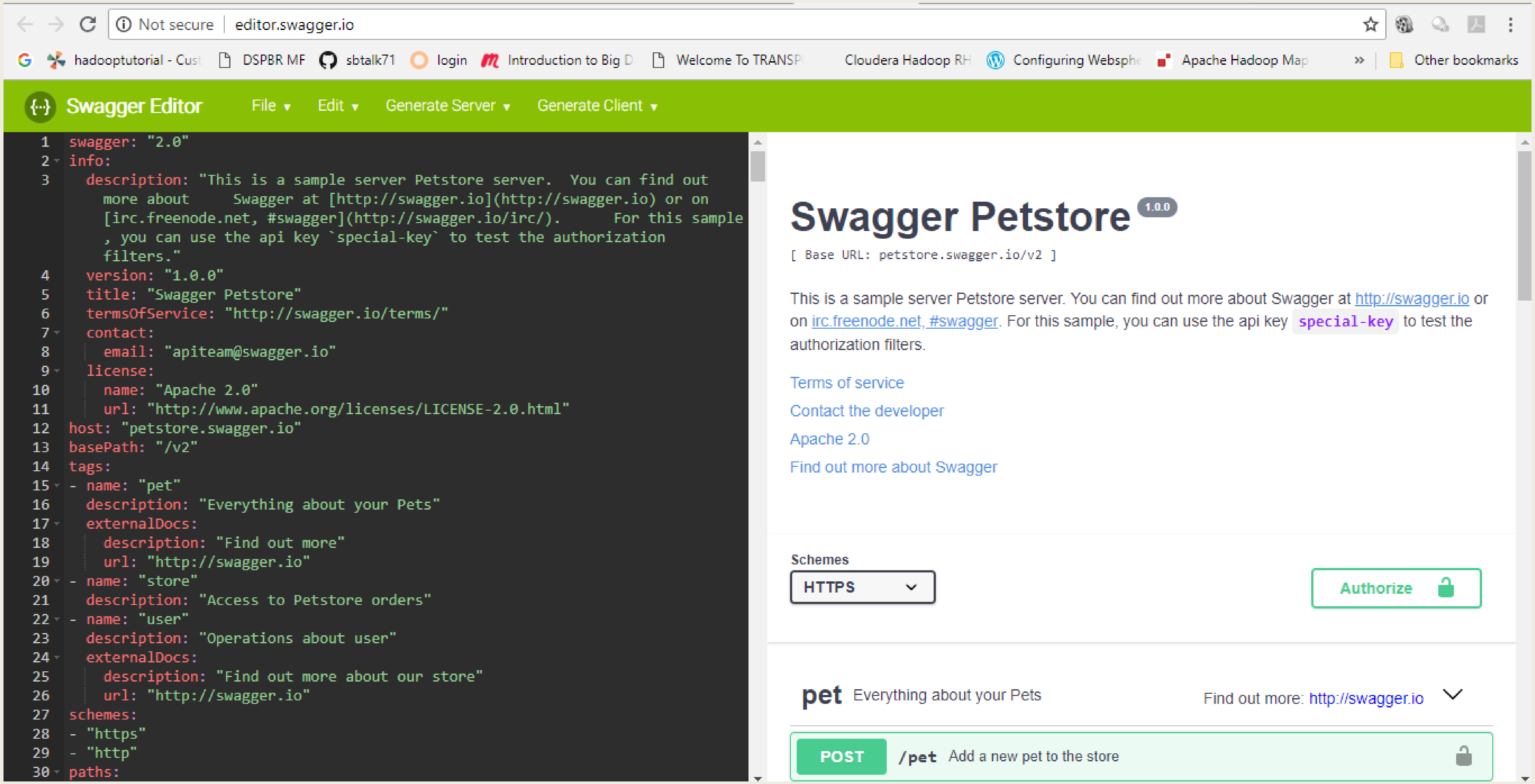
- The Swagger representation of the API is made of a single OpenAPI Specification document
- Represented as JSON OR YAML
- All field names are case sensitive
- Primitive data types in the Swagger (OpenAPI) Specification are based on the types supported by the **JSON-Schema Draft 4**.
- Models are described using the Schema Object which is a subset of JSON Schema Draft 4.

Sample OpenAPI Document

```
paths:
  /pet:
    post:
      tags:
        - pet
      summary: Add a new pet to the store
      x-swagger-router-controller: SampleController
      description: ""
      operationId: addPet
      consumes:
        - application/json
        - application/xml
      produces:
        - application/xml
        - application/json
      parameters:
        - in: body
          name: body
          description: Pet object that needs to be added to the store
          required: false
```

```
schema:
          $ref: "#/definitions/Pet"
     responses:
       "405":
         description: Invalid input
     security:
       - petstore_auth:
           - "write:pets"
           - "read:pets"
```

Swagger PetStore Document



```
swagger: "2.0"
info:
  description: "This is a sample server Petstore server.  You can find out
    more about     Swagger at [http://swagger.io](http://swagger.io) or on
    [irc.freenode.net, #swagger](http://swagger.io/irc/).      For this sample
    , you can use the api key `special-key` to test the authorization
    filters."
  version: "1.0.0"
  title: "Swagger Petstore"
  termsOfService: "http://swagger.io/terms/"
  contact:
    email: "apiteam@swagger.io"
  license:
    name: "Apache 2.0"
    url: "http://www.apache.org/licenses/LICENSE-2.0.html"
host: "petstore.swagger.io"
basePath: "/v2"
tags:
- name: "pet"
  description: "Everything about your Pets"
  externalDocs:
    description: "Find out more"
    url: "http://swagger.io"
- name: "store"
  description: "Access to Petstore orders"
- name: "user"
  description: "Operations about user"
  externalDocs:
    description: "Find out more about our store"
    url: "http://swagger.io"
schemes:
- "https"
- "http"
paths:
```

Swagger Petstore 1.0.0

[Base URL: petstore.swagger.io/v2]

This is a sample server Petstore server. You can find out more about Swagger at http://swagger.io or on irc.freenode.net, #swagger. For this sample, you can use the api key `special-key` to test the authorization filters.

Terms of service

Contact the developer

Apache 2.0

Find out more about Swagger

Schemes

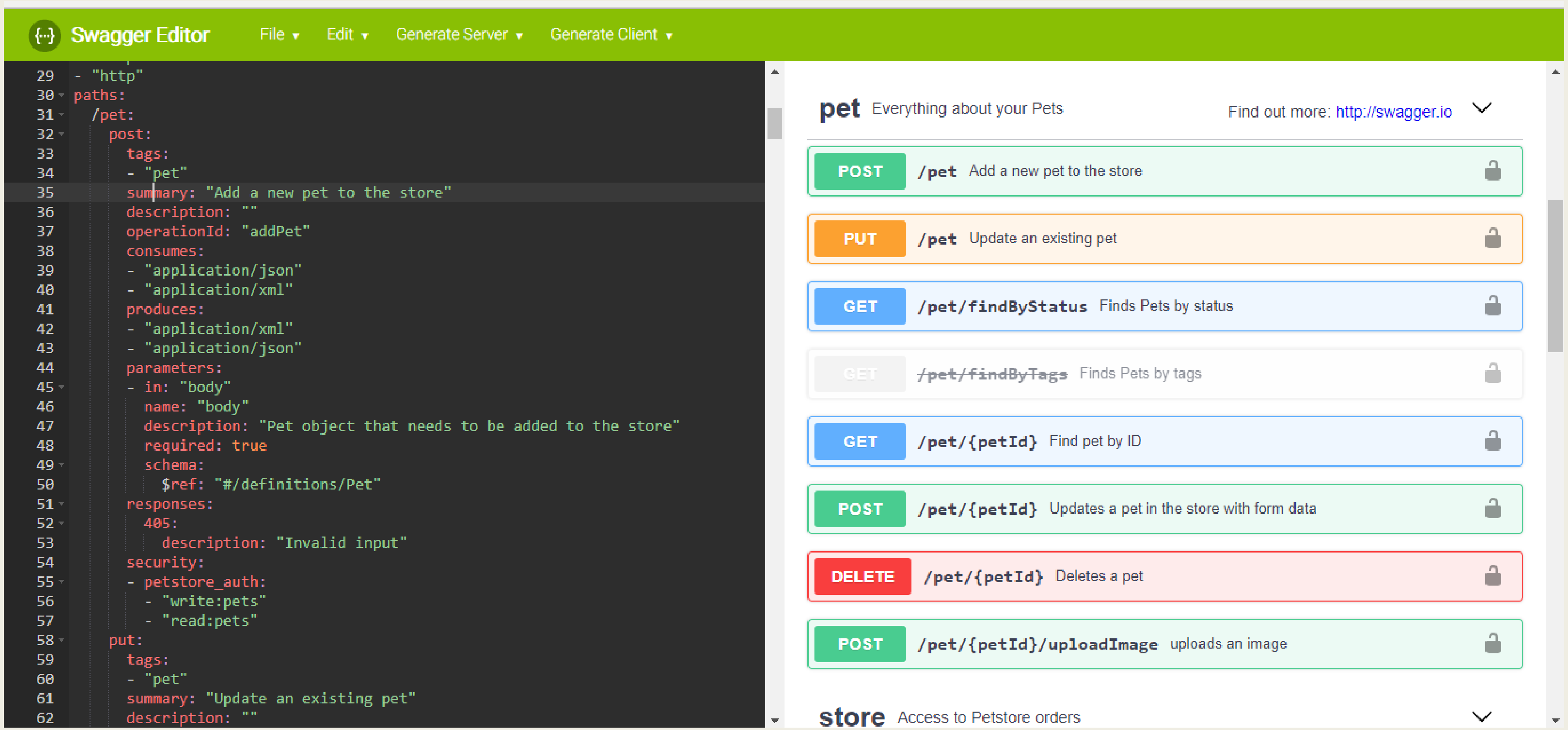
HTTPS

Authorize

pet Everything about your Pets — Find out more: http://swagger.io

POST /pet Add a new pet to the store

Swagger PetStore Document



```yaml
- "http"
paths:
  /pet:
    post:
      tags:
      - "pet"
      summary: "Add a new pet to the store"
      description: ""
      operationId: "addPet"
      consumes:
      - "application/json"
      - "application/xml"
      produces:
      - "application/xml"
      - "application/json"
      parameters:
      - in: "body"
        name: "body"
        description: "Pet object that needs to be added to the store"
        required: true
        schema:
          $ref: "#/definitions/Pet"
      responses:
        405:
          description: "Invalid input"
      security:
      - petstore_auth:
        - "write:pets"
        - "read:pets"
    put:
      tags:
      - "pet"
      summary: "Update an existing pet"
      description: ""
```

pet Everything about your Pets — Find out more: http://swagger.io

POST `/pet` Add a new pet to the store

PUT `/pet` Update an existing pet

GET `/pet/findByStatus` Finds Pets by status

GET ~~`/pet/findByTags`~~ Finds Pets by tags

GET `/pet/{petId}` Find pet by ID

POST `/pet/{petId}` Updates a pet in the store with form data

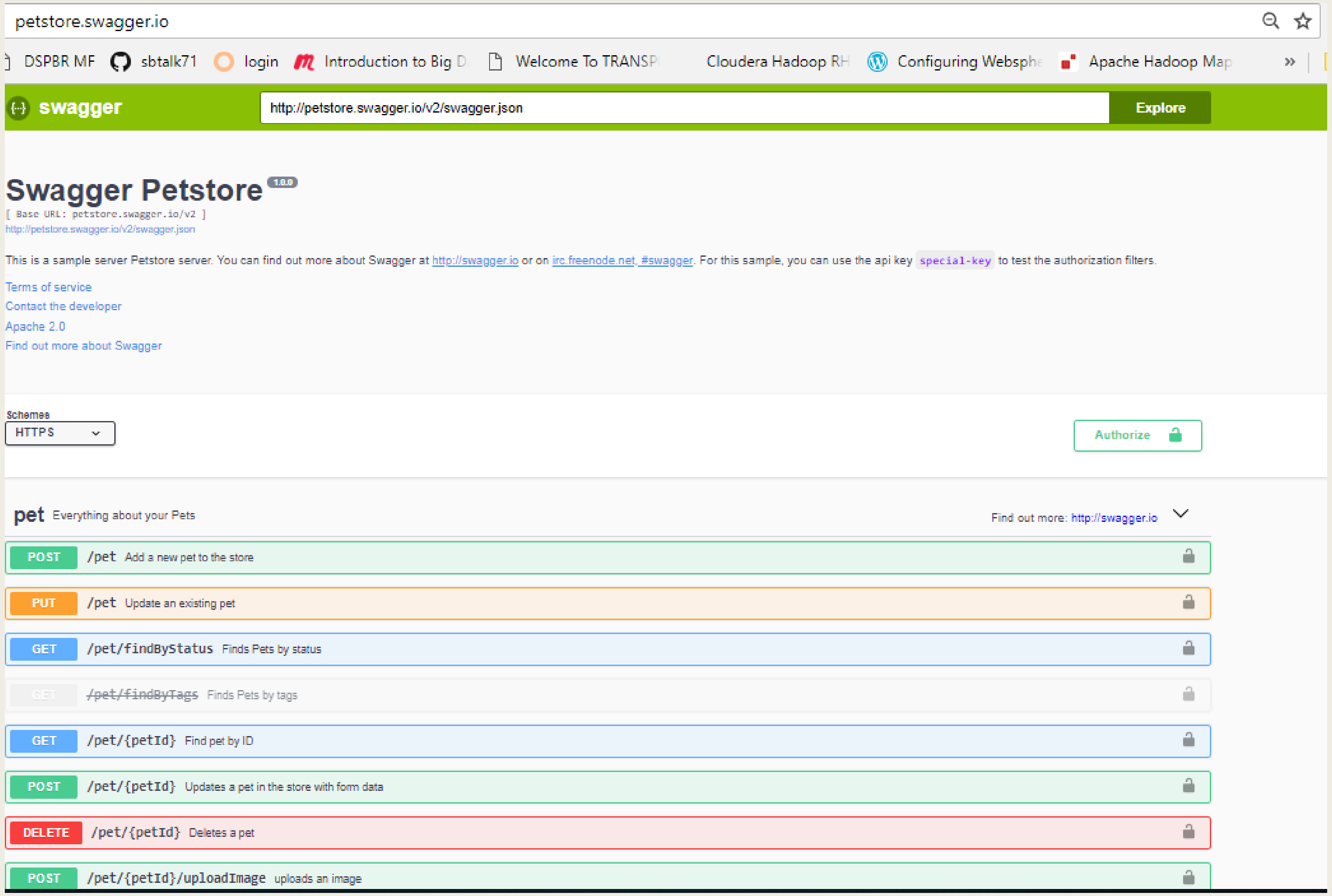
DELETE `/pet/{petId}` Deletes a pet

POST `/pet/{petId}/uploadImage` uploads an image

store Access to Petstore orders

Swagger UI

- Swagger UI provides a display framework that reads an OpenAPI specification document and generates an interactive documentation website.



Swagger Petstore 1.0.0

[Base URL: petstore.swagger.io/v2]
http://petstore.swagger.io/v2/swagger.json

This is a sample server Petstore server. You can find out more about Swagger at http://swagger.io or on irc.freenode.net, #swagger. For this sample, you can use the api key `special-key` to test the authorization filters.

Terms of service
Contact the developer
Apache 2.0
Find out more about Swagger

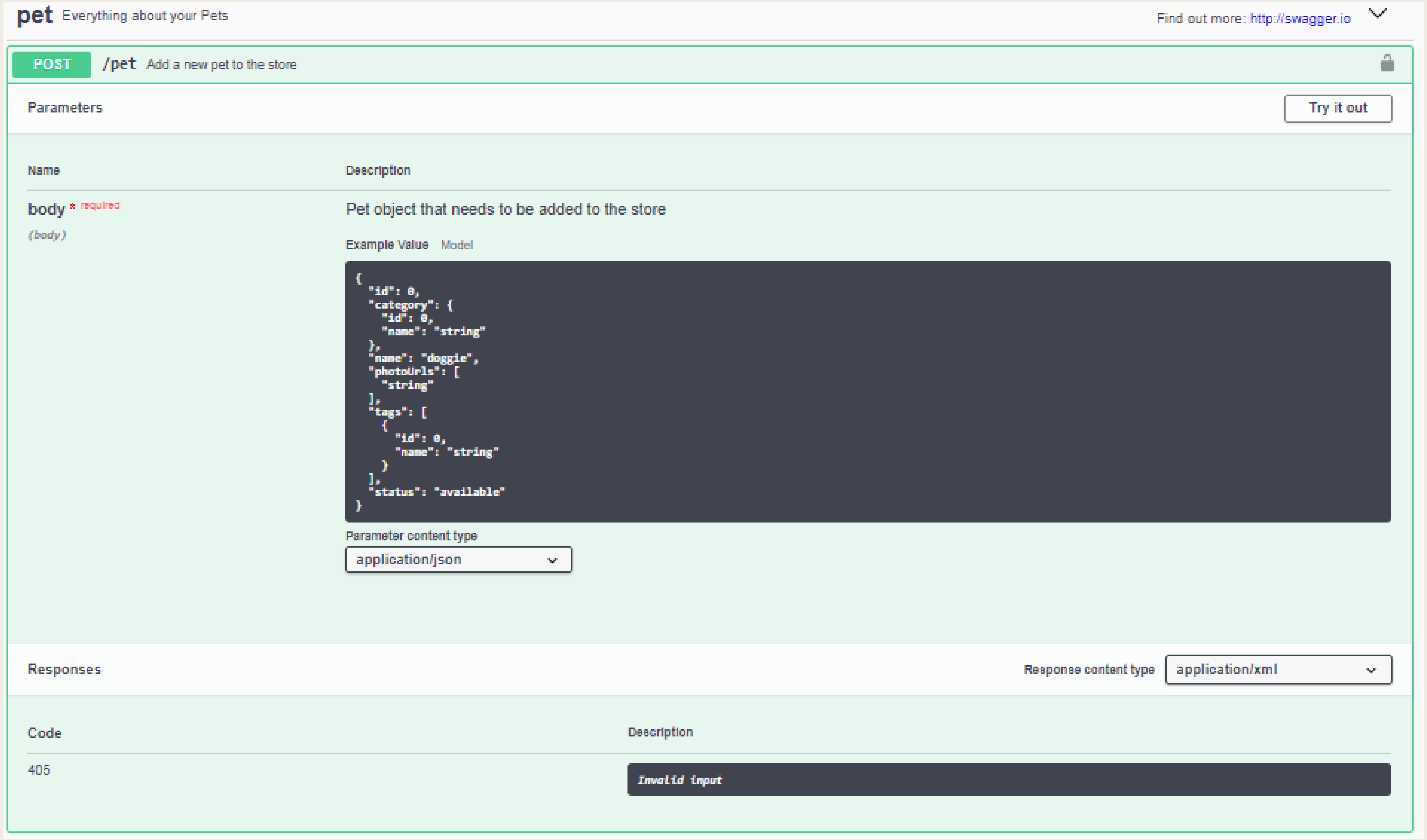
Schemes: HTTPS

Authorize

pet Everything about your Pets — Find out more: http://swagger.io

POST `/pet` Add a new pet to the store
PUT `/pet` Update an existing pet
GET `/pet/findByStatus` Finds Pets by status
GET ~~`/pet/findByTags`~~ Finds Pets by tags
GET `/pet/{petId}` Find pet by ID
POST `/pet/{petId}` Updates a pet in the store with form data
DELETE `/pet/{petId}` Deletes a pet
POST `/pet/{petId}/uploadImage` uploads an image

Swagger UI Example



pet Everything about your Pets

Find out more: http://swagger.io

POST /pet Add a new pet to the store

Parameters

Try it out

Name	Description
body * required *(body)*	Pet object that needs to be added to the store

Example Value Model

```
{
  "id": 0,
  "category": {
    "id": 0,
    "name": "string"
  },
  "name": "doggie",
  "photoUrls": [
    "string"
  ],
  "tags": [
    {
      "id": 0,
      "name": "string"
    }
  ],
  "status": "available"
}
```

Parameter content type: application/json

Responses

Response content type: application/xml

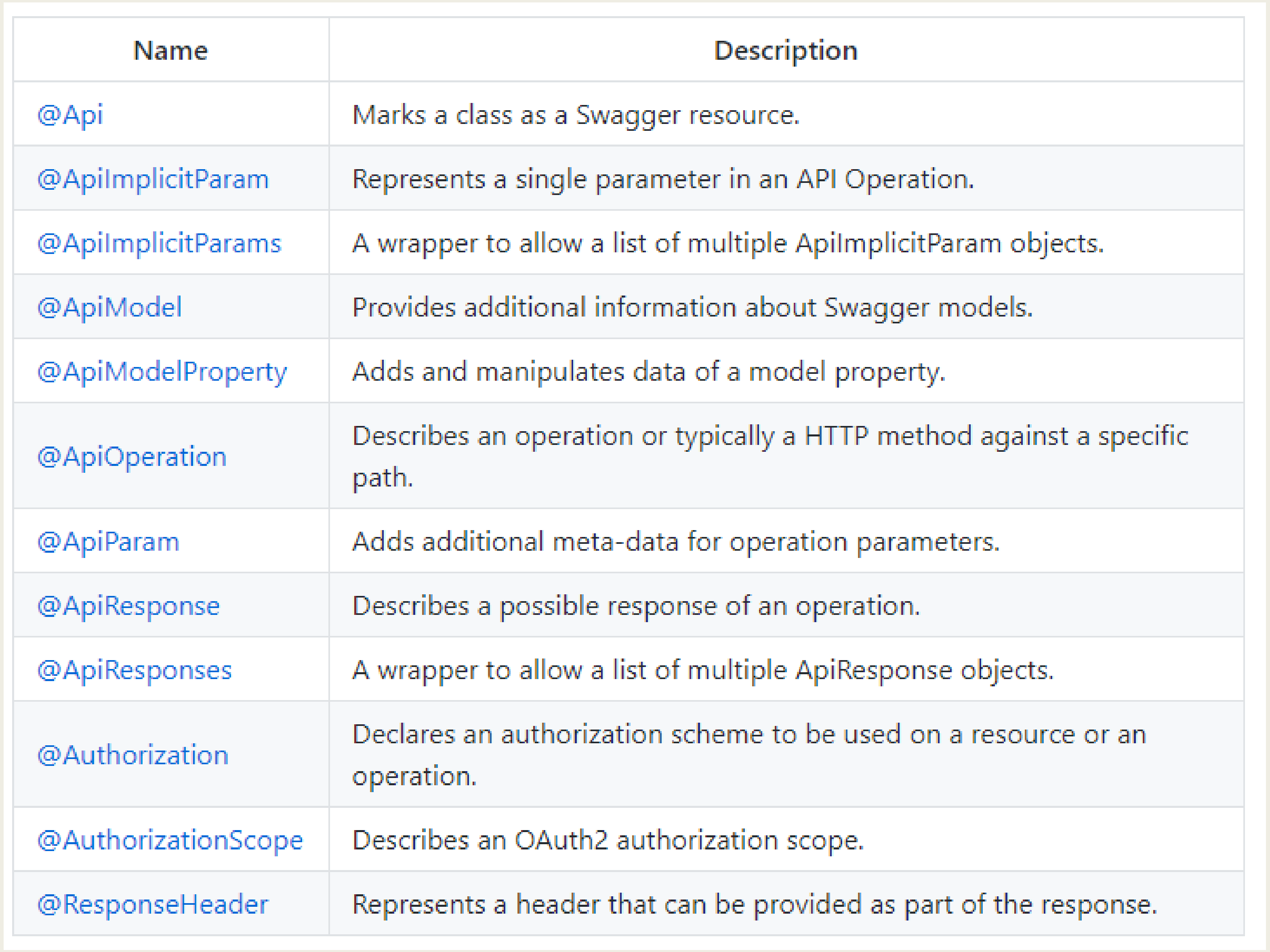
Code	Description
405	*Invalid input*

COMMUNITY-DRIVEN LANGUAGE INTEGRATIONS

- Clojure
- ColdFusion / CFML
- Eiffel
- Go
- Groovy
- Java
- JavaScript
- Node.js
- Perl
- PHP
- Python
- Ruby
- Scala

SWAGGER WITH JAVA

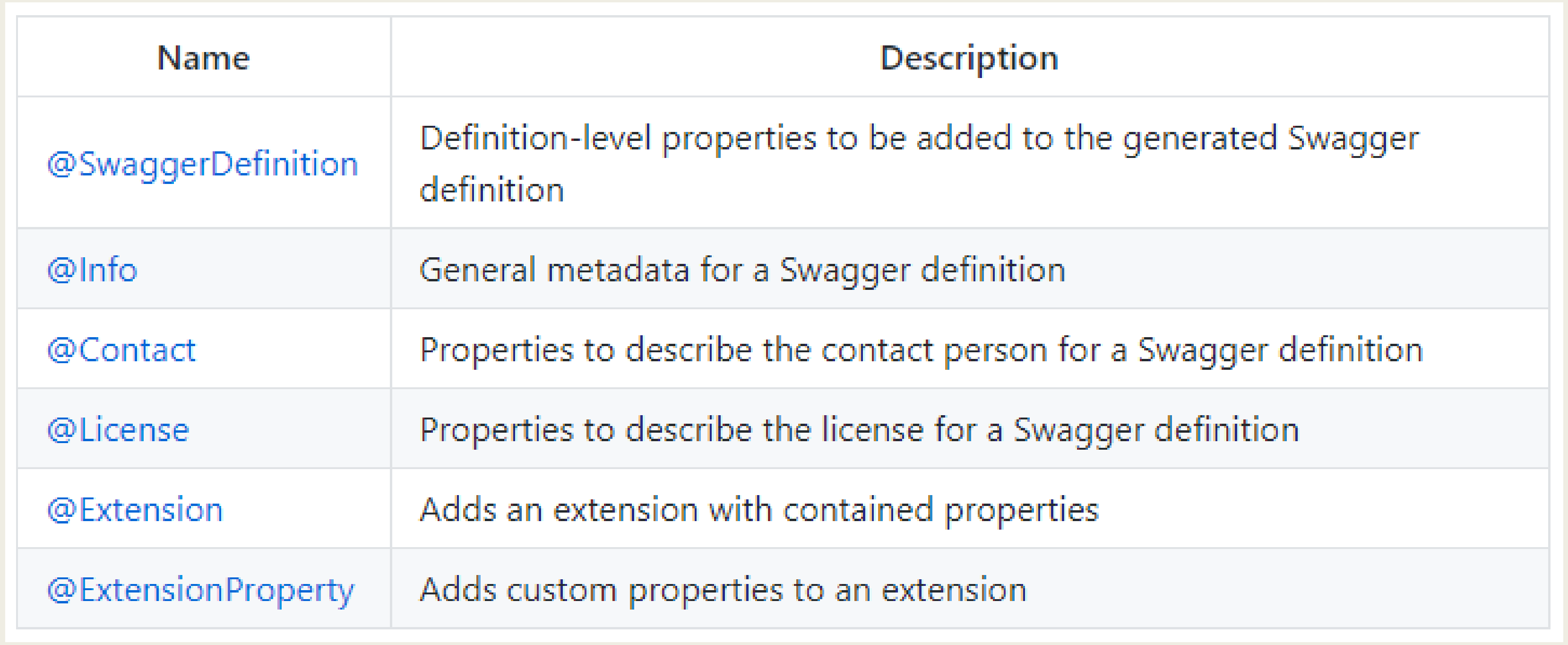
Java Annotations in Swagger



Name	Description
@Api	Marks a class as a Swagger resource.
@ApiImplicitParam	Represents a single parameter in an API Operation.
@ApiImplicitParams	A wrapper to allow a list of multiple ApiImplicitParam objects.
@ApiModel	Provides additional information about Swagger models.
@ApiModelProperty	Adds and manipulates data of a model property.
@ApiOperation	Describes an operation or typically a HTTP method against a specific path.
@ApiParam	Adds additional meta-data for operation parameters.
@ApiResponse	Describes a possible response of an operation.
@ApiResponses	A wrapper to allow a list of multiple ApiResponse objects.
@Authorization	Declares an authorization scheme to be used on a resource or an operation.
@AuthorizationScope	Describes an OAuth2 authorization scope.
@ResponseHeader	Represents a header that can be provided as part of the response.

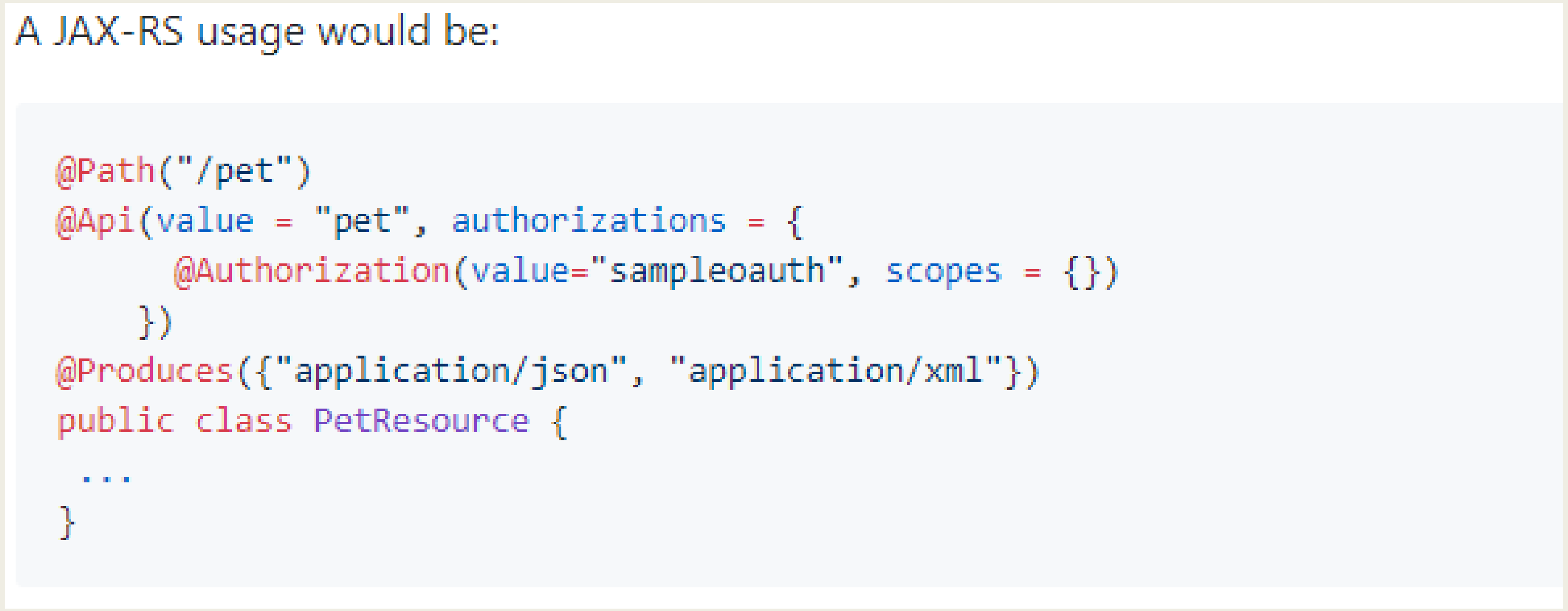
Java Annotations in Swagger

The latest release also adds a number of annotations for adding extensions and metadata at the Swagger Definition level:



Name	Description
@SwaggerDefinition	Definition-level properties to be added to the generated Swagger definition
@Info	General metadata for a Swagger definition
@Contact	Properties to describe the contact person for a Swagger definition
@License	Properties to describe the license for a Swagger definition
@Extension	Adds an extension with contained properties
@ExtensionProperty	Adds custom properties to an extension

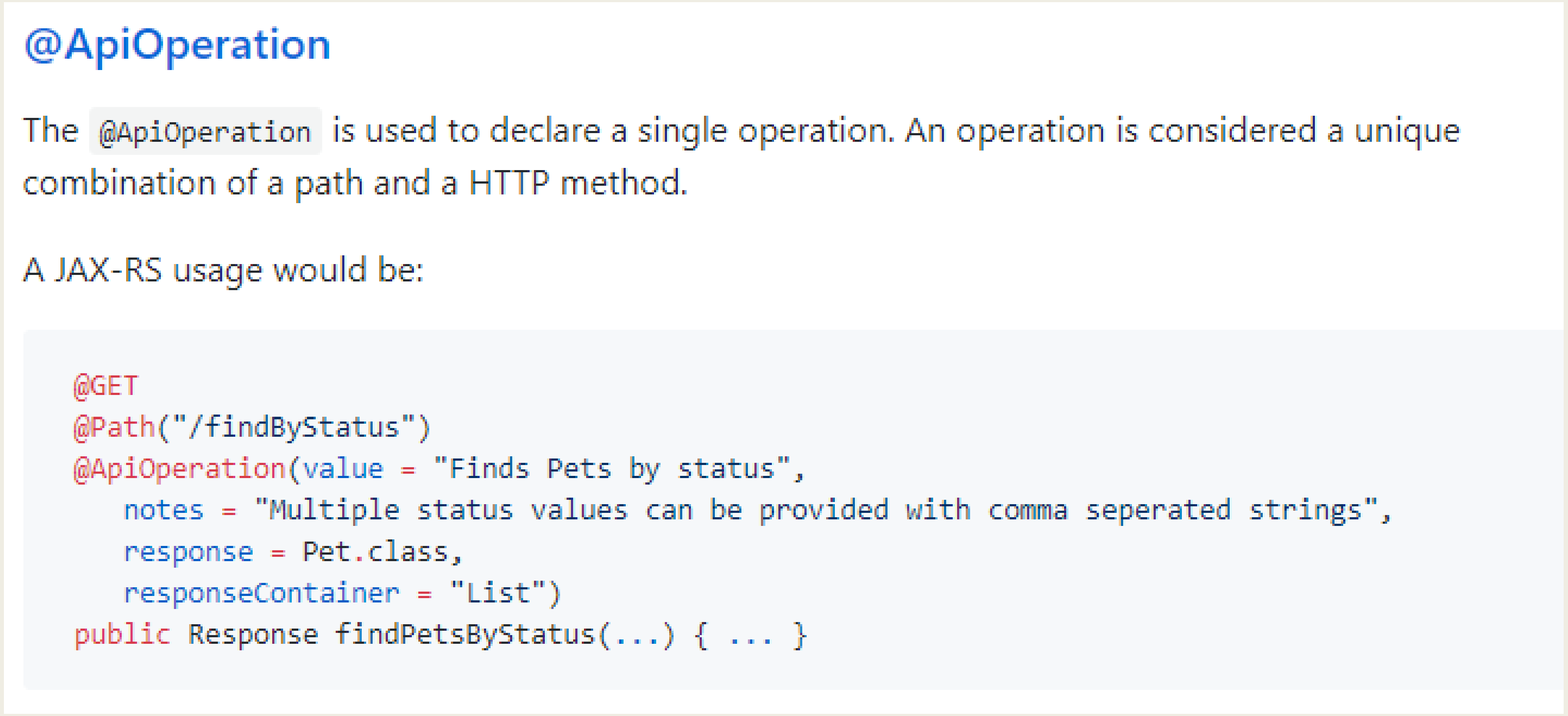
@Api



A JAX-RS usage would be:

```
@Path("/pet")
@Api(value = "pet", authorizations = {
      @Authorization(value="sampleoauth", scopes = {})
    })
@Produces({"application/json", "application/xml"})
public class PetResource {
 ...
}
```

@ApiOperation
(Operation Declaration)



@ApiOperation

The `@ApiOperation` is used to declare a single operation. An operation is considered a unique combination of a path and a HTTP method.

A JAX-RS usage would be:

```
@GET
@Path("/findByStatus")
@ApiOperation(value = "Finds Pets by status",
   notes = "Multiple status values can be provided with comma seperated strings",
   response = Pet.class,
   responseContainer = "List")
public Response findPetsByStatus(...) { ... }
```

All Other Annotations

For Greater Details on other annotations refer to:

https://github.com/swagger-api/swagger-core/wiki/Annotations-1.5.X

SWAGGER SUPPORT IN SPRING BOOT

Swagger with Spring

- In Spring Framework, we use **Springfox** implementation of the Swagger specification.
- Add the maven dependency as given below

```
<dependency>
    <groupId>io.springfox</groupId>
    <artifactId>springfox-swagger2</artifactId>
    <version>2.7.0</version>
</dependency>
```

Swagger UI

- To Include Swagger UI in your Spring application add the following dependency

```
<dependency>
    <groupId>io.springfox</groupId>
    <artifactId>springfox-swagger-ui</artifactId>
    <version>2.7.0</version>
</dependency>
```

Swagger Configuration

```
@Configuration
@EnableSwagger2
@ComponentScan(basePackageClasses=GreetController.class)
public class SwaggerConfig {

	public ApiInfo apiInfo() {
		return new ApiInfoBuilder().license("Apache 2.0 License")
					.description("A Demo Application")
					.title("Greeter App").build();
	}

	@Bean
	public Docket productApi() {
		return new Docket(DocumentationType.SWAGGER_2)
				.select()
				.apis(RequestHandlerSelectors.basePackage("com.demo.spring"))
				 .paths(PathSelectors.regex("/app.*"))
				.build()
				.apiInfo(apiInfo());
	}
}
```

The Controller Class (Our Minimal REST API)

```
@RestController
@Api(value = "GreeterApp")
@RequestMapping("/app")
public class GreetController {

        @ApiOperation(value="greets a named person")
        @GetMapping(path = "/greet", produces = "text/plain")
        public String greet(@RequestParam("name") String name) {
                return "Welcome To Swagger " + name;
        }
}
```

QUESTIONS?

THANK YOU