富士通 Java ブログ

MicroProfile OpenAPI による API 管理（Part1）：既存の Java コードをドキュメント化する

2020 年 5 月 15 日 初版
数村 憲治

はじめに

マイクロサービスを始めたときに出てくる課題の一つに、サービスを追加・修正する度に必要となる API 管理があります。このシリーズでは 2 回に分けて、MicroProfile OpenAPI を使った API 管理（注 1）を紹介します。
なお、ここで紹介するプログラムの完全なソースコードは、以下で参照できます。

- https://github.com/fujitsu/app_blog/tree/master/openapi-202003/part1（GitHub, Inc.）

注 1　ここで言う「API 管理」とは、いわゆる、サービスディスカバリーのようなことではありません。

マイクロサービスでの API 管理

マイクロサービスアーキテクチャーでシステムを構築する際は、一般的にはサービス間を API でつないでいきます。サービス提供者は、API の仕様を、サービス利用者に提供します。サービス利用者は、サービスの中身について知る必要はなく、提供された API の仕様にしたがって呼出しさえすれば、サービスを利用することができます。したがって、重要なのは、サービス提供者が、サービスの内容に合致した正しい仕様を提供することです。また、サービス内容を更新し、API に変更が生じる場合は、タイムリーにサービス利用者に変更した仕様を提供することです。

新しくサービスを作成する際には、まじめに仕様書を作ることが多いですが、サービスを更新したときに、同時に仕様書も更新することは億劫になりがちです。また、サービスの数が少ないときはいいですが、マイクロサービスアーキテクチャーのように、数千、数万のサービスを管理するのはたいへんです。さらに、年月が経つと、当時開発した人が現場からいなくなり、仕様書もない状態で誰もメンテナンスできない、ということもあります。

OpenAPI

OpenAPI Specification とは、REST API の仕様を定義するための標準ルールで、特定のプログラミング言語に依存しない仕様となっています。また、仕様書は YAML や JSON 形式で記述可能なため、仕様書をプログラム的に扱うことができます。そのため、人手で管理する必要がなくなり、API の数がいくら増えても、管理コストを抑えることができます。

OpenAPI では、さまざまなツールが開発され利用できますが、本シリーズでは、Swagger UI（OpenAPI 形式の仕様書を表示するツール）と、OpenAPI Generator（OpenAPI 形式の仕様書から HTML やプログラムコードなどを作成するツール）を使います。

MicroProfile OpenAPI

MicroProfile OpenAPI とは、OpenAPI を Java のインターフェイスで使うもので、JAX-RS アプリケーションから OpenAPI 形式のドキュメントを作成することができます。

MicroProfile OpenAPI で用意しているさまざまな annotation を JAX-RS のコードに付与することで、OpenAPI の仕様書を作成できますが、第 1 回では、これらの annotation は一切使用せずに、素の JAX-RS コードだけで、仕様書を作成します。すなわち、想定としては、すでに既存の動作している JAX-RS プログラムがあり、それらのコードを修正することなく、API 仕様書を作成します。

既存の JAX-RS から OpenAPI 仕様書を作成する

ソースの準備

使用する JAX-RS のプログラムは以下のような HelloWorld です。

App.java

```
package com.fujitsu.launcher.demo202003;
import javax.ws.rs.ApplicationPath;
import javax.ws.rs.core.Application;

@ApplicationPath("/demo")
public class App extends Application {
}
```

Hello.java

```
package com.fujitsu.launcher.demo202003;
import javax.ws.rs.GET;
import javax.ws.rs.Path;

@Path("hello")
public class Hello {

  @GET
  public String hello() {
    return "Hello World !";
  }
}
```

プログラムのビルドについては、こちらに maven プロジェクトを用意していますので、参照ください。

HelloWorld の起動

MicroProfile OpenAPI を使用するために、MicroProfile の実装の一つである、Launcher を用意します。ここでは、以下より、2.0-b01 版をダウンロードして利用します。

- https://github.com/fujitsu/launcher/releases/download/2.0-b01/launcher-2.0-b01.jar（GitHub, Inc.）

Launcher の使い方は、ダウンロードした「launcher-2.0-b01.jar」を任意の場所に置いて、java コマンドの-jar オプションに指定する（インストール作業不要）だけです。詳細な Launcher の使用方法は、ドキュメントを参照してください。

```
$ java -jar launcher-2.0-b01.jar --deploy Hello.war
```

通常、MicroProfile アプリケーションの起動はこれでいいのですが、今回は、仕様書を生成したいパッケージ名を mp.openapi.scan.packages プロパティに指定して仕様書を作成します。

```
$ java -Dmp.openapi.scan.packages=com.fujitsu.launcher.demo202003 -jar launcher-2.0-b01.jar --deploy Hello.war
```


仕様書の確認

アプリケーションはこれまで通り、設定したエンドポイントにアクセスできます。

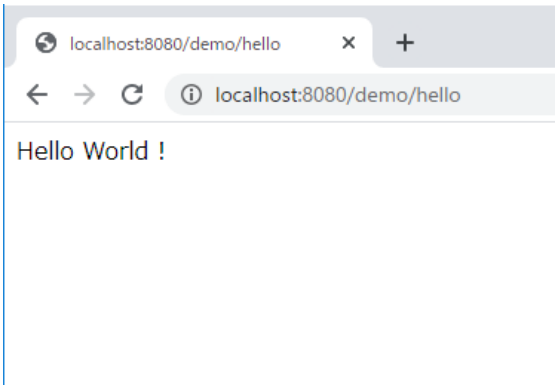

図 1

OpenAPI 形式の仕様は、「/openapi」でアクセスすることで取得可能です。

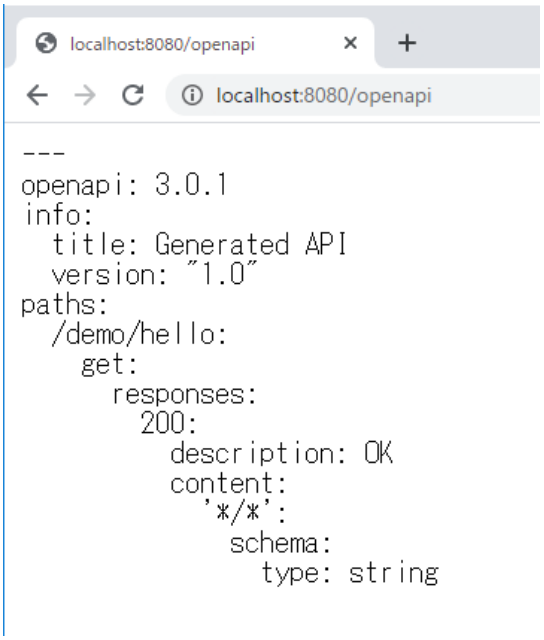

図 2

HTML の作成

JSON や YAML 形式は、プログラム的に扱うには都合が良いですが、人が目で見るにはいささか見難くなります。そこで、OpenAPI Generator を使い、YAML 形式の仕様書から、HTML を生成します。OpenAPI Generator は、GitHub で開発されており、バイナリーは Maven Central から取得することができます。今回は、バージョン 4.2.3 を使用します。以下のように、-i オプションに、OpenAPI 形式の仕様を指定し、-g オプションに「html」を指定することで、HTML を生成することができます。

```
$ java -jar openapi-generator-cli-4.2.3.jar generate -i http://localhost:8080/openapi -g html
```

カレントディレクトリーに、index.html ができるので、それをブラウザーで開きます。

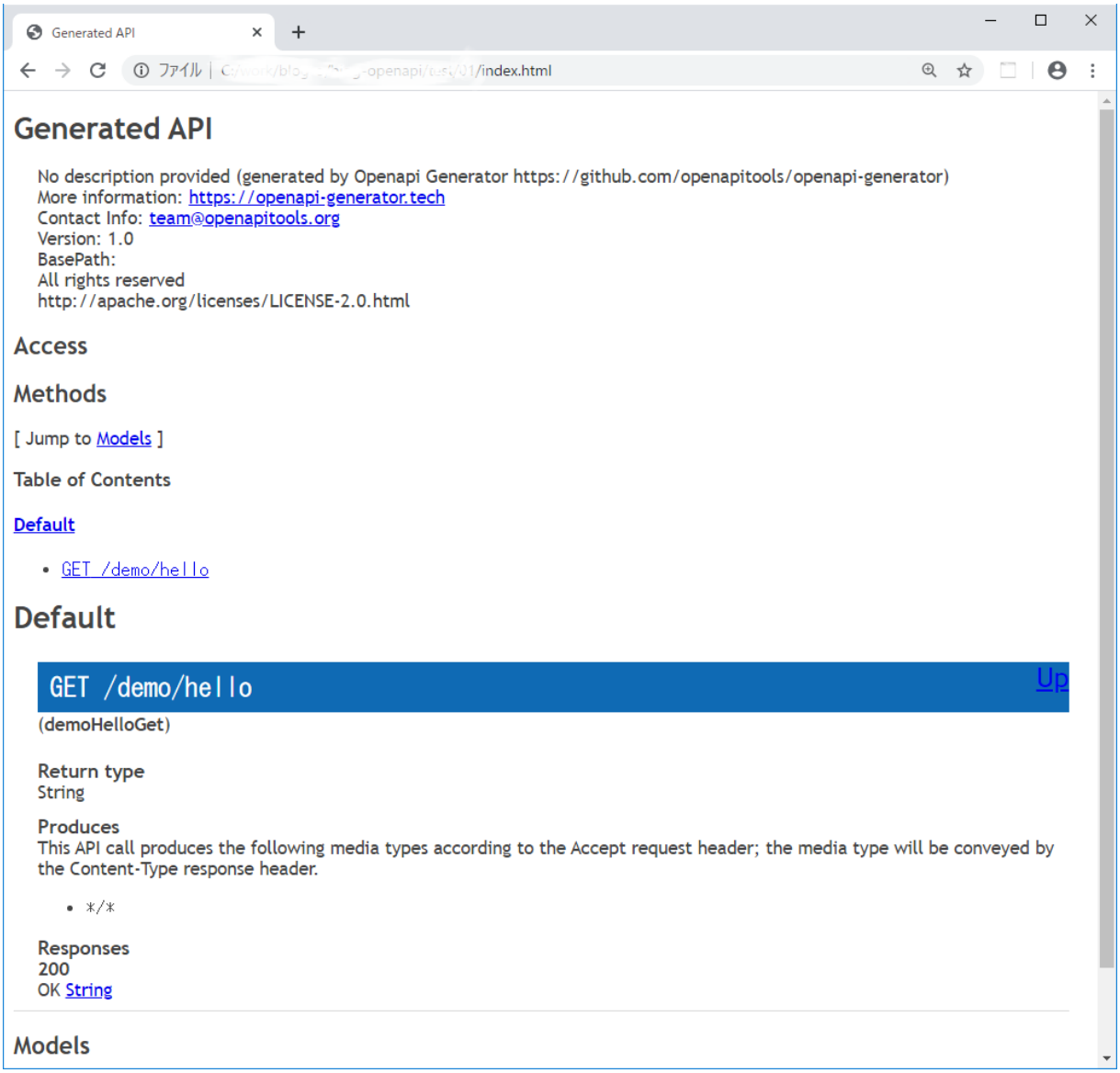

図 3

このように、限定的な情報ではありますが、API のエンドポイントごとに、return type やレスポンスヘッダーの情報などを自動生成することができます。

Swagger UI

HTML にすることで、だいぶ見やすくなりましたが、generator を使って、いちいち HTML に変換するのも面倒です。そこで、MicroProfile の拡張機能を使い、自動的に Swagger UI に対応させてみます。MicroProfile の拡張機能を使うには、以下の dependency を pom.xml に追加します。

```
<dependency>
  <groupId>org.microprofile-ext.openapi-ext</groupId>
  <artifactId>openapi-ui</artifactId>
  <version>1.1.2</version>
</dependency>
```

あとは、これまでと同様に maven でビルドします。Swagger UI へのアクセスは、「<アプリケーションのベース URI>/openapi-ui」でアクセスします。

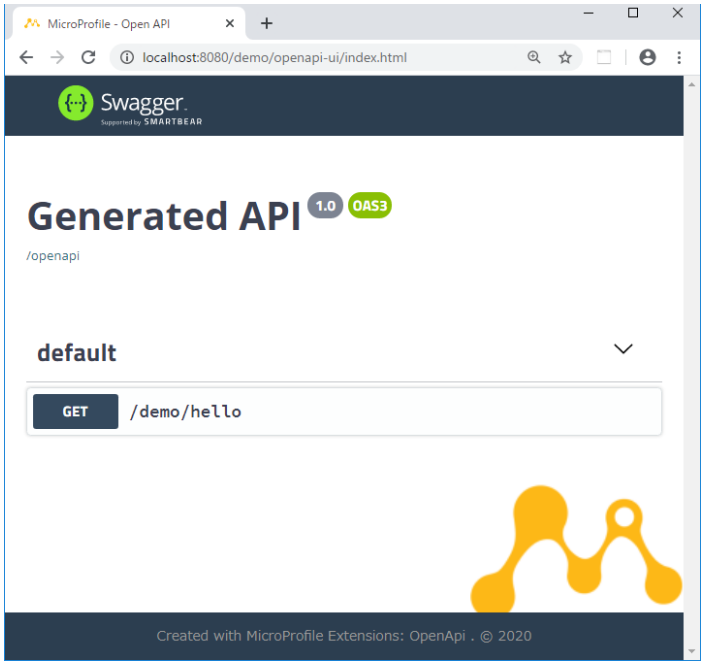

図 4

「/demo/hello」のところをクリックすると、以下のような詳細情報を見ることができます。

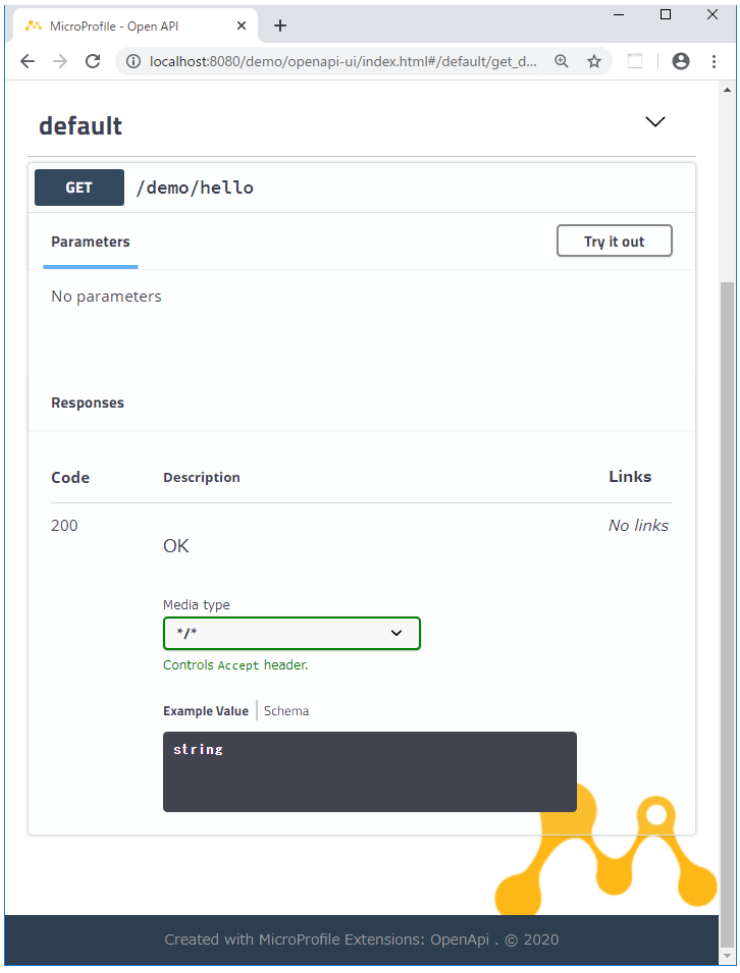

図 5

まとめと予告

今回 Part1 では、MicroProfile OpenAPI の annotation は一切使わず、ドキュメントを生成することができました。このことは、すでに存在する JAX-RS のプログラムの中身を見なくても、ある程度のドキュメントを生成し、メンテナンスに役立てることができることを意味します。

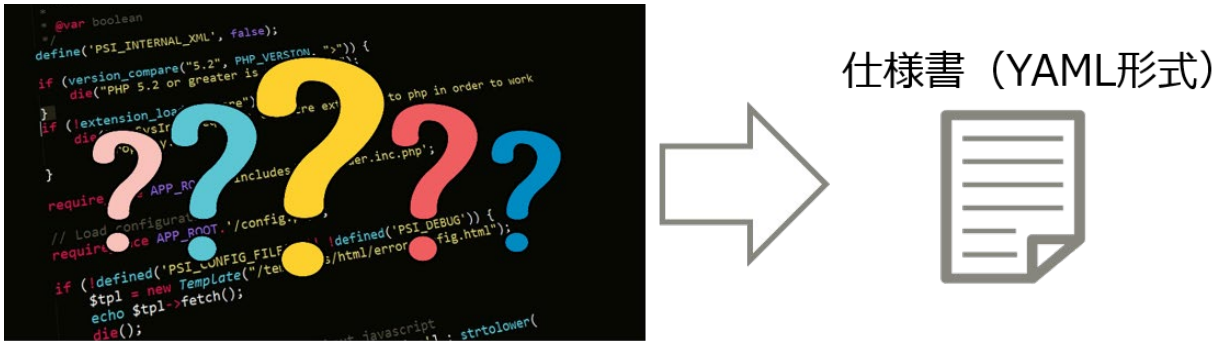

図 6

次回 Part2 では、MicroProfile OpenAPI を使って、より詳細なドキュメントを作成する方法を見ていきます。