



# Experience Swagger

---

HIKARI ZHEN 2020/12/24



# SWAGGERとは

RESTful APIのドキュメントや、サーバ、クライアントコード、エディタ、またそれらを扱うための仕様などを提供するフレームワークです。

「The World's Most Popular Framework for APIs」と言われています。

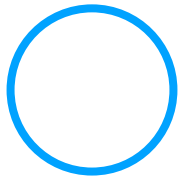





# なぜ **SWAGGER**

変化し続ける要件に対応するために、API経由で連携するマイクロサービスのアーキテクチャが増えてきています。

従来のモノリシックなアプリケーションについて、機能間のインターフェイスをどう管理するかという話が来ました。





# 基本的なツール

## **swagger-core**

JavaのコードからSwagger Specificationを生成するためのJavaライブラリ

<https://github.com/swagger-api/swagger-core>

## **Swagger Editor**

Swagger Specificationからクライアントコード生成するコマンドラインツール

<https://editor.swagger.io/>





# 基本的なツール

## Swagger Codegen

Swagger Specificationからクライアントコード生成するコマンドラインツール

<https://github.com/swagger-api/swagger-codegen>

## Swagger UI

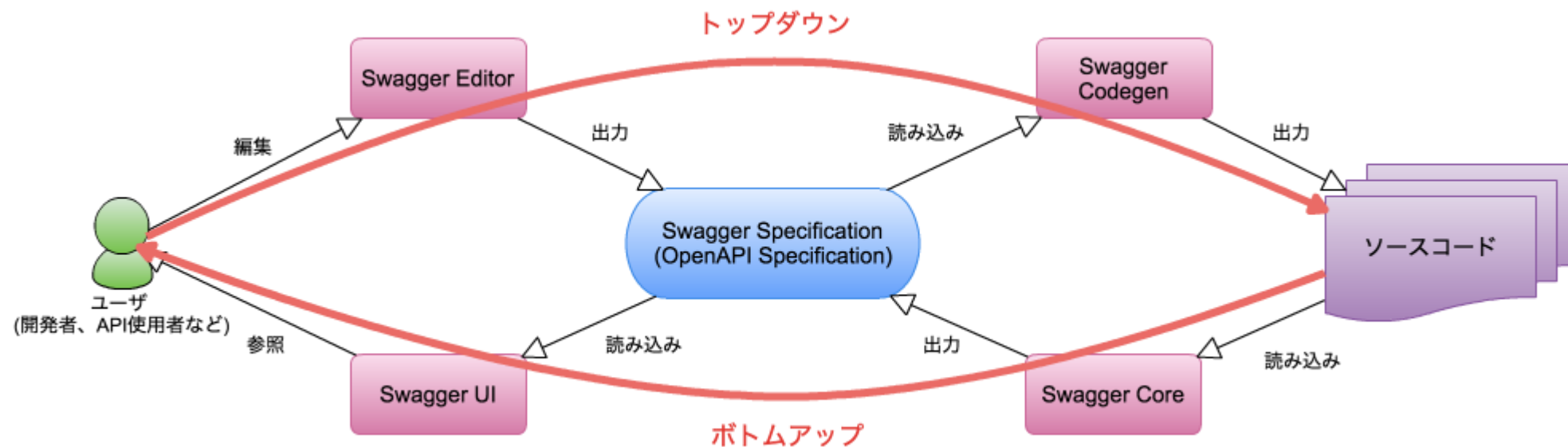
Swagger Specificationから動的にドキュメントを生成するツール

<https://swagger.io/tools/swagger-ui/>



# どう使うのか

Swaggerにて、トップダウン形式とボトムアップ形式があります。



# 「HELLO WORLD」例

## 「Swagger Editor」でAPI定義作業を行う

The screenshot displays the Swagger Editor interface. On the left, a code editor shows a Swagger 2.0 definition for a 'Greeting Server sample'. The definition includes a 'hello' endpoint with a 'get' method that produces 'application/json' and has a query parameter 'name' with a default value. On the right, the visual representation of the API is shown, including a 'default' tab, a 'GET /hello' endpoint, a 'Parameters' section with a 'name' query parameter, and a 'Responses' section with a '200' response that returns a greeting.

```
1 swagger: '2.0'
2 info:
3   version: 1.0.0
4   title: "Greeting Server sample"
5 paths:
6   /hello:
7     get:
8       produces:
9         - "application/json"
10      parameters:
11        - name: name
12          required: false
13          type: "string"
14          in: "query"
15          description: "defaults to HelloWorld if not given"
16      operationId: "getGreeting"
17      responses:
18        '200':
19          description: "returns a greeting"
20          schema:
21            type: "string"
22            description: "contains the actual greeting as plain text"
23
```

default

GET /hello

Parameters

Try it out

Name	Description
name	defaults to HelloWorld if not given

string (query)

name - defaults to HelloWorld if not given

Responses

Response content type: application/json

Code	Description
200	returns a greeting

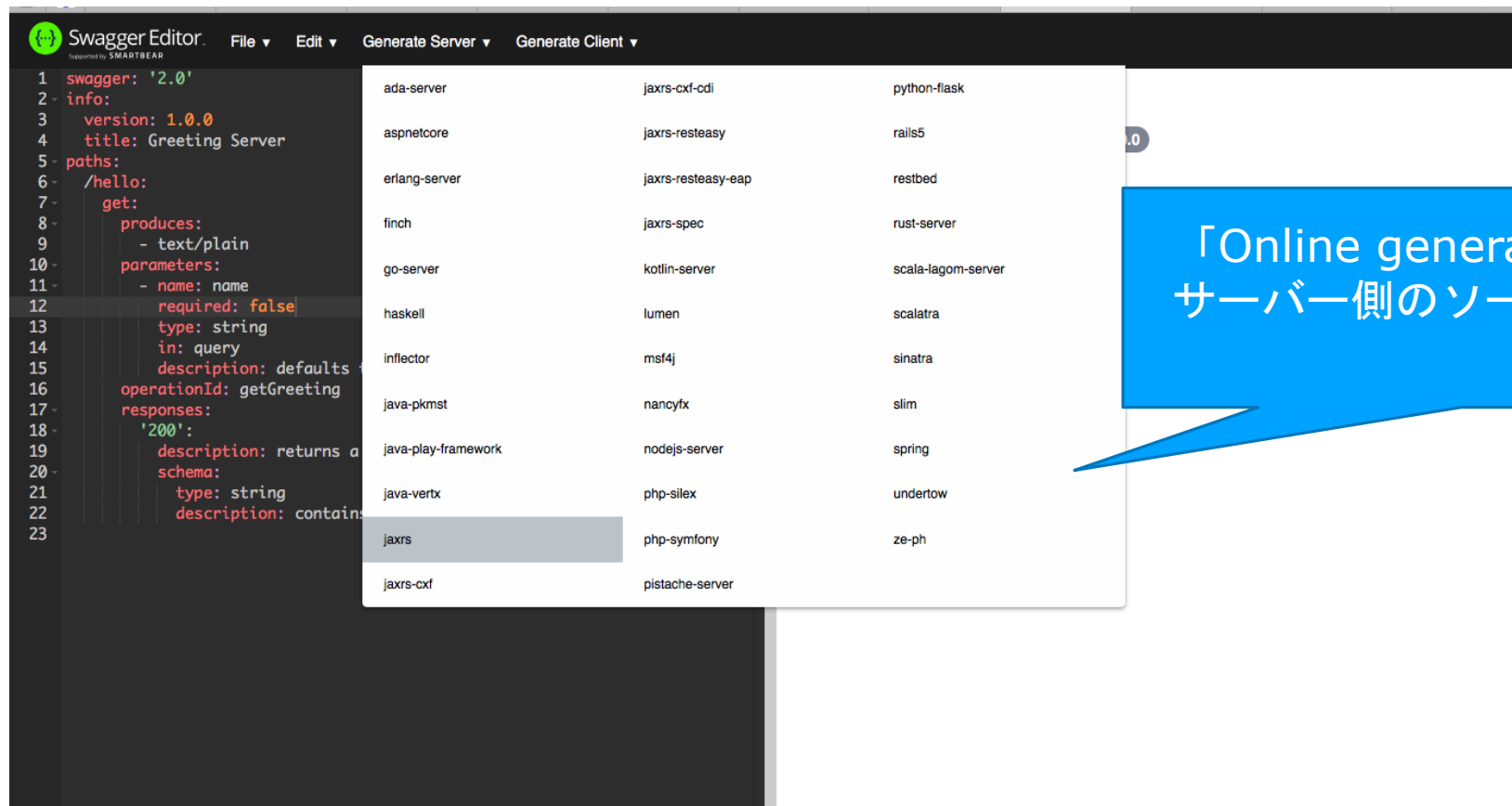
Example Value | Model

"string"

ドキュメントは自動的に生成される

# 「HELLO WORLD」例

「Swagger Codegen」を利用してソースを自動生成します。



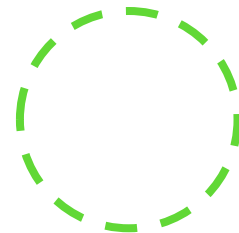
The screenshot shows the Swagger Editor interface. On the left, a Swagger specification is displayed in a code editor. On the right, a dropdown menu lists various server-side code generators. The 'jaxrs' generator is highlighted.

```
1 swagger: '2.0'
2 info:
3   version: 1.0.0
4   title: Greeting Server
5 paths:
6   /hello:
7     get:
8       produces:
9         - text/plain
10      parameters:
11        - name: name
12          required: false
13          type: string
14          in: query
15          description: defaults
16      operationId: getGreeting
17      responses:
18        '200':
19          description: returns a
20          schema:
21            type: string
22            description: contains
```

ada-server	jaxrs-cxf-cdi	python-flask
aspnetcore	jaxrs-resteasy	rails5
erlang-server	jaxrs-resteasy-eap	restbed
finch	jaxrs-spec	rust-server
go-server	kotlin-server	scala-lagom-server
haskell	lumen	scalatra
inflector	mst4j	sinatra
java-pkms	nancyfx	slim
java-play-framework	nodejs-server	spring
java-vertx	php-silex	undertow
jaxrs	php-symfony	ze-ph
jaxrs-cxf	pistache-server	

「Online generators」を利用して  
サーバー側のソースを生成します。





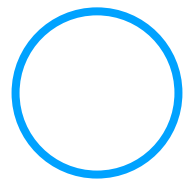
# 「HELLO WORLD」例

生成したソースは zip ファイルの形でダウンロードします。

社内開発であれば、セキュリティ視点から「swagger-codegen」をプロジェクトに導入する。

勉強する時、オンライン生成は可能です。

※オンライン生成ガイドライン <https://github.com/swagger-api/swagger-codegen#online-generators>



# 「HELLO WORLD」例

生成したソースに対して仮実装を行う

```
63 63 + public Response getGreeting(@ApiParam(value = "defaults to HelloWorld if not given") @QueryParam("name")
64 64 , @Context SecurityContext securityContext)
65 65 throws NotFoundException {
66 - return delegate.getGreeting(name, securityContext);
66 + String p_name = name;
67 + if (null == name) {
68 +     p_name = "";
69 + }
70 + if ("".equals(p_name)) {
71 +     p_name = "hello swagger";
72 + } else {
73 +     p_name = "hello " + p_name;
74 + }
75 + return delegate.getGreeting(p_name, securityContext);
67 76 }
68 77 }
```

# 「HELLO WORLD」例

生成したソースをコンパイルする。

\$ mvn clean package jetty:run

```
jaxrs-server — java -classpath /usr/local/Cellar/maven/3.6.3.1/libexec/boot/plexus-classworlds-2.6.0.jar -Dclassworlds.conf=/usr/local/Cellar/maven/3.6.3.1/libexec/bin/m2.conf -Dmaven.home=/usr/lo...
...sr/local/Cellar/maven/3.6.3.1/libexec/lib/jansi-native -Dmaven.multiModuleProjectDirectory=/Users/jinghuizhen/project/swagger/gen/jaxrs-server org.codehaus.plexus.classworlds.launcher.Launcher clean package jetty:run
Downloaded from central: https://repo.maven.apache.org/maven2/org/mortbay/jasper/apache-el/8.0.33/apache-el-8.0.33.jar (241 kB at 92 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/javax/transaction/javax.transaction-api/1.2/javax.transaction-api-1.2.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/eclipse/jetty/apache-jstl/9.3.27.v20190418/apache-jstl-9.3.27.v20190418.jar (3.7 kB at 1.4 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/taglibs/taglibs-standard-spec/1.2.5/taglibs-standard-spec-1.2.5.jar (40 kB at 15 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/eclipse/jdt/core/compiler/ecj/4.4.2/ecj-4.4.2.jar (2.3 MB at 802 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/taglibs/taglibs-standard-impl/1.2.5/taglibs-standard-impl-1.2.5.jar (206 kB at 72 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/javax/transaction/javax.transaction-api/1.2/javax.transaction-api-1.2.jar (31 kB at 11 kB/s)
[INFO] Logging initialized @93767ms
[INFO] Configuring Jetty for project: swagger-jaxrs-server
[INFO] webAppSourceDirectory/Users/jinghuizhen/project/swagger/gen/jaxrs-server/target/swagger-jaxrs-server-1.0.0 does not exist. Trying src/main/webapp
[INFO] Reload Mechanic: automatic
[INFO] Classes = /Users/jinghuizhen/project/swagger/gen/jaxrs-server/target/classes
[INFO] Context path = /
[INFO] Tmp directory = /Users/jinghuizhen/project/swagger/gen/jaxrs-server/target/tmp
[INFO] Web defaults = org.eclipse.jetty/webapp/webdefault.xml
[INFO] Web overrides = none
[INFO] web.xml file = file:///Users/jinghuizhen/project/swagger/gen/jaxrs-server/src/main/webapp/WEB-INF/web.xml
[INFO] Webapp directory = /Users/jinghuizhen/project/swagger/gen/jaxrs-server/src/main/webapp
[INFO] jetty-9.3.27.v20190418, build timestamp: 2019-04-19T03:11:38+09:00, git hash: d3e249f86955d04bc646bb620905b7c1bc596a8d
[INFO] Scanning elapsed time=575ms
[INFO] Started o.e.j.m.p.JettyWebAppContext@5c997de8{/,file:///Users/jinghuizhen/project/swagger/gen/jaxrs-server/src/main/webapp/,AVAILABLE}{file:///Users/jinghuizhen/project/swagger/gen/jaxrs-server/src/main/webapp/}
[INFO] Started ServerConnector@6e069ca9[HTTP/1.1,[http/1.1]]{0.0.0.0:8080}
[INFO] Started @96033ms
[INFO] Started Jetty Server
13:13:19.527 [qtp52562984-40] DEBUG io.swagger.jaxrs.ext.SwaggerExtensions - adding extension io.swagger.jersey.SwaggerJersey2Jaxrs@2b2c14a4
13:13:19.558 [qtp52562984-40] DEBUG io.swagger.converter.ModelConverterContextImpl - resolveProperty class java.lang.String
13:13:19.559 [qtp52562984-40] DEBUG io.swagger.jackson.ModelResolver - resolveProperty [simple type, class java.lang.String]
13:13:19.591 [qtp52562984-40] DEBUG io.swagger.converter.ModelConverterContextImpl - resolveProperty class java.lang.String
13:13:19.591 [qtp52562984-40] DEBUG io.swagger.jackson.ModelResolver - resolveProperty [simple type, class java.lang.String]
13:13:19.592 [qtp52562984-40] DEBUG io.swagger.jaxrs.Reader - getParameters for [simple type, class java.lang.String]
13:13:19.592 [qtp52562984-40] DEBUG io.swagger.jaxrs.Reader - trying extension io.swagger.jersey.SwaggerJersey2Jaxrs@2b2c14a4
13:13:19.594 [qtp52562984-40] DEBUG io.swagger.converter.ModelConverterContextImpl - resolveProperty [simple type, class java.lang.String]
13:13:19.594 [qtp52562984-40] DEBUG io.swagger.jackson.ModelResolver - Can't check class [simple type, class java.lang.String], java.lang.String
13:13:19.594 [qtp52562984-40] DEBUG io.swagger.jackson.ModelResolver - resolveProperty [simple type, class java.lang.String]
13:13:19.609 [qtp52562984-40] DEBUG io.swagger.jaxrs.Reader - getParameters for [simple type, class javax.ws.rs.core.SecurityContext]
13:13:19.610 [qtp52562984-40] DEBUG io.swagger.jaxrs.Reader - trying extension io.swagger.jersey.SwaggerJersey2Jaxrs@2b2c14a4
13:13:19.610 [qtp52562984-40] DEBUG io.swagger.jaxrs.Reader - no parameter found, looking at body params
13:13:19.610 [qtp52562984-40] DEBUG io.swagger.jaxrs.Reader - trying to decorate operation: io.swagger.jersey.SwaggerJersey2Jaxrs@2b2c14a4
13:13:19.614 [qtp52562984-40] DEBUG io.swagger.jaxrs.listing.BaseApiListingResource - configuring swagger with io.swagger.jaxrs.config.WebXMLReader@3b0d7ec9
12月 17, 2020 1:14:43 午後 org.glassfish.jersey.message.internal.WriterInterceptorExecutor$TerminalWriterInterceptor aroundWriteTo
重大: MessageBodyWriter not found for media type=text/plain, type=class io.swagger.api.ApiResponseMessage, genericType=class io.swagger.api.ApiResponseMessage.
12月 17, 2020 1:18:16 午後 org.glassfish.jersey.message.internal.WriterInterceptorExecutor$TerminalWriterInterceptor aroundWriteTo
重大: MessageBodyWriter not found for media type=text/plain, type=class io.swagger.api.ApiResponseMessage, genericType=class io.swagger.api.ApiResponseMessage.
```

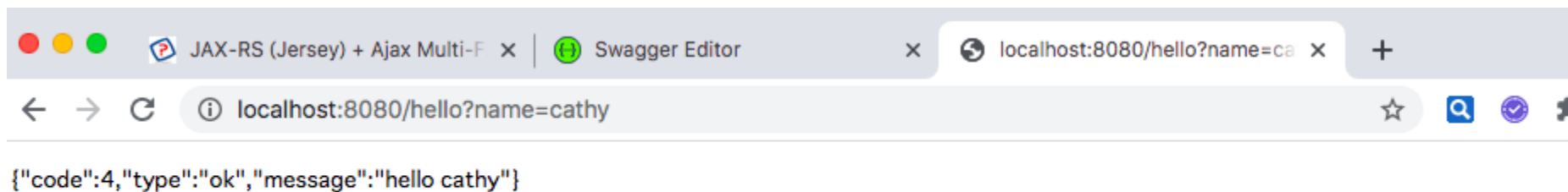
# 「HELLO WORLD」例

## 結果を確認する

### コマンドで確認

```
$ curl http://localhost:8080/hello?name=Cathy {"code":4,"type":"ok","message":"hello Cathy"}
```

### ブラウザで確認



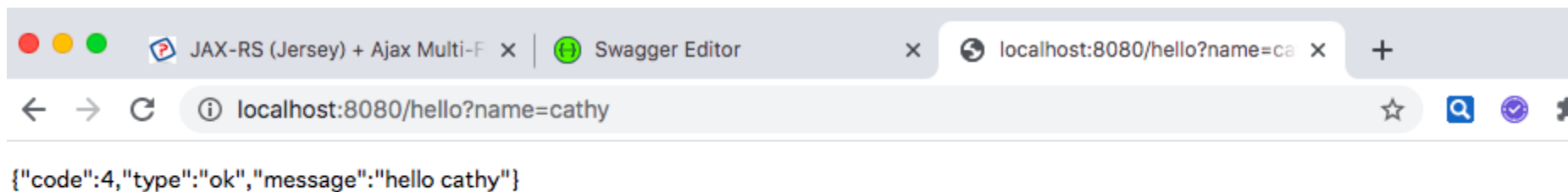
# 「HELLO WORLD」例

## 結果を確認する

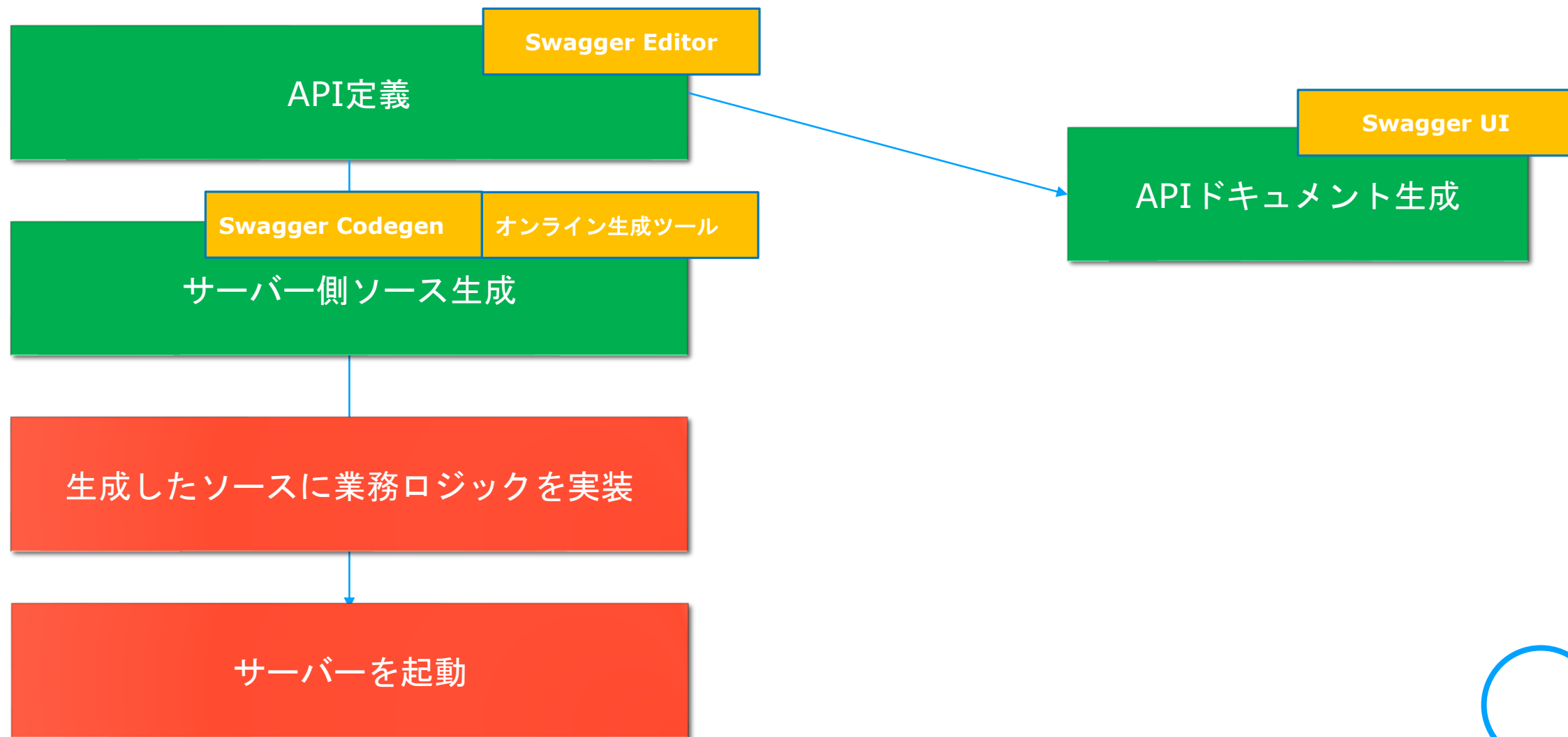
### コマンドで確認

```
$ curl http://localhost:8080/hello?name=Cathy {"code":4,"type":"ok","message":"hello Cathy"}
```

### ブラウザで確認



# 「HELLO WORLD」例まとめ



# 「FILE API」例

/file/{fileId}

ファイル情報を取得する。

リターン値

FWFileオブジェクト

(JSON形)

添付したYamlファイルを

ご参照してください。

The image shows a side-by-side comparison of a REST API definition. On the left is the Swagger Editor interface with a dark theme, displaying a YAML file for a GET endpoint. On the right is a light-themed UI representation of the same API.

**Swagger Editor (Left):**

```
24 /file/{fileId}:
25   get:
26     tags:
27       - "file"
28     summary: "Find file by ID"
29     description: "Returns a file information"
30     operationId: "getFileById"
31     produces:
32       - "application/xml"
33       - "application/json"
34     parameters:
35       - name: "fileId"
36         in: "path"
37         description: "ID of file to return"
38         required: true
39         type: "string"
40     responses:
41       "200":
42         description: "successful operation"
43         schema:
44           $ref: "#/definitions/FWFile"
45       "400":
46         description: "Invalid ID supplied"
47       "404":
48         description: "File not found"
49   definitions:
50     FWFile:
51       type: "object"
52       required:
53         - "name"
54         - "photoUrls"
55       properties:
56         id:
57           type: "string"
58           example: "20201217001"
59         name:
60           type: "string"
61           example: "swagger intro"
62         status:
63           type: "string"
64           description: "file status"
```

**API UI (Right):**

**file**

**GET /file/{fileId}** Find file by ID

Returns a file information

**Parameters** Try it out

Name	Description
<b>fileId</b> * required	ID of file to return
string (path)	fileId - ID of file to return

**Responses** Response content type: application/xml

Code	Description
200	successful operation
400	Invalid ID supplied

**Example Value | Model**

```
<?xml version="1.0" encoding="UTF-8"?>
<FWFile>
  <id>20201217001</id>
  <name>swagger intro</name>
  <status>available</status>
</FWFile>
```



# 「FILE API」例

業務仮実装概要

{fileId}が

「 1 」： 仮のファイル情報を返却する。

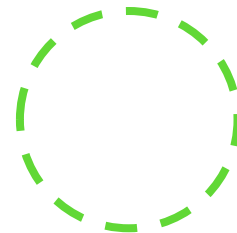
「 2 」： ファイルは存在しない情報を返却する。

「 3 」： エラーが発生した情報を返却する。



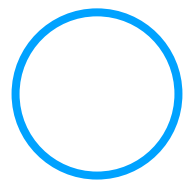


# 「FILE API」例



17 gen\_v2/jaxrs-server/src/main/java/io/swagger/api/impl/FileApiServiceImpl.java

```
@@ -20,6 +20,21 @@
20      @Override
21      public Response getFileById(String fileId, SecurityContext securityContext) throws NotFoundException {
22          // do some magic!
23          - return Response.ok().entity(new ApiResponseMessage(ApiResponseMessage.OK, "magic!")).build();
23          + System.out.println(">>>FileApiServiceImpl#getFileById fileId=" + fileId);
24          +
25          + if ("1".equals(fileId)) {
26          +     FWFile file = new FWFile();
27          +     file.setId("20201220001");
28          +     file.setName("file_sample");
29          +     file.setStatus(FWFile.StatusEnum.AVAILABLE);
30          +
31          +     System.out.println(">>>FileApiServiceImpl#getFileById simulate file is generated.");
32          +
33          +     return Response.ok().entity(file).build();
34          + } else if ("2".equals(fileId)) {
35          +     return Response.ok().entity(new ApiResponseMessage(ApiResponseMessage.WARNING, "file is not found.")).build();
36          + } else {
37          +     return Response.ok().entity(new ApiResponseMessage(ApiResponseMessage.ERROR, "some error is raised.")).build();
38          + }
24      }
25  }
```





# 「FILE API」例



検証：

```
$ curl http://localhost:8080/file/1
```

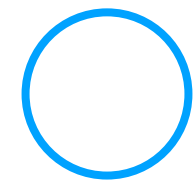
```
{"id":"20201220001","name":"file_sample","status":"available"}
```

```
$ curl http://localhost:8080/file/2
```

```
{"code":2,"type":"warning","message":"file is not found."}
```


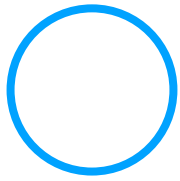
```
$ curl http://localhost:8080/file/3
```

```
{"code":1,"type":"error","message":"some error is raised."}
```





# IMFとの結合方式

1. Swaggerプロジェクトを作成する。
  2. File操作APIを定義する。
  3. サーバー側ソースを自動生成する。
  4. IMFをSwaggerプロジェクトに導入する。
  5. Swaggerプロジェクトをデプロイする。
- 
- 



# 添付ファイル

■ HelloWorldサンプル API定義ファイル

helloworld.yml

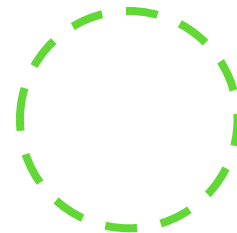
■ ファイル操作 API定義ファイル

fileApi.yml





# SWAGGER調査報告



ありがとうございました。

