openSAP SAP Cloud Platform API Management
Additional Download

openSAP
open.sap.com

THE BEST RUN

TABLE OF CONTENTS

In this exercise, you'll explore the use of API first design and show how OpenAPI and the SAP API Designer can help develop your first micro service. The microservice you'll create would be for product catalog and product details like stock availability and price information.

SYSTEM PREREQUISITES

- Account on the SAP Cloud Platform
- Node JS version 6.12.3 or later installed (http://www.nodejs.org)
- Node Package Manager (NPM) 3.10 or later installed (should be automatically installed when you install Node JS)
- Visual Studio Code (https://code.visualstudio.com/) or another code editor

API GUIDELINES AND BEST PRACTICES

SAP-defined API harmonization guidelines enable customers and partners to build APIs to offer to other consumers. Gain practical guidance by putting the consumer experience at the center of design decisions.

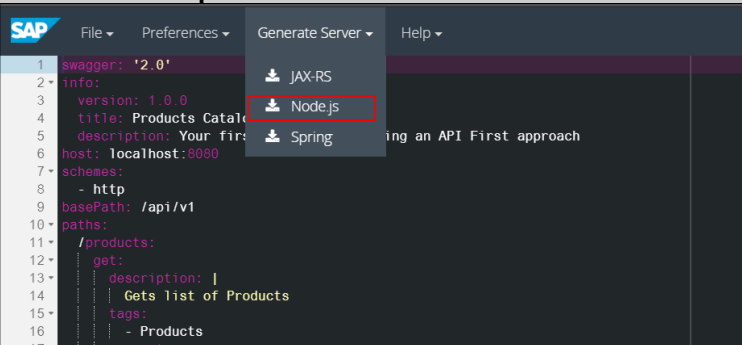For more details, you can go through the SAP Guidelines for REST API Harmonization.

API DESIGN

Open API Specification

Step	Screenshot/ Sample
Product catalog OpenAPI sample productcatalog.yaml	swagger: '2.0' info: version: 1.0.0 title: Products Catalog API description: Your first microservice using an API First approach host: localhost:8080 schemes: - http basePath: /api/v1 paths: /products: get: description: | Gets list of Products tags: - Products parameters: - $ref: '#/parameters/limitParam' - $ref: '#/parameters/offsetParam' responses: '200': description: List of Products schema: type: array items: $ref: '#/definitions/product' '500': description: Server error schema: $ref: '#/definitions/error' post: description: | Create Product tags: - Products parameters: - in: body name: payload description: Product payload schema: $ref: '#/definitions/product'

```yaml
          required: true
        responses:
          '201':
            description: Product created Successfully
            schema:
              $ref: '#/definitions/product'
          '500':
            description: Server error
            schema:
              $ref: '#/definitions/error'
  '/products/{id}':
    delete:
      description: Delete the selected product
      tags:
        - Products
      parameters:
        - name: id
          type: string
          in: path
          required: true
      responses:
        '204':
          description: Product deleted Successfully
        '404':
          description: Item not found
          schema:
            $ref: '#/definitions/error'
        '500':
          description: Server error
          schema:
            $ref: '#/definitions/error'
    get:
      description: Get the product by id
      tags:
        - Products
      parameters:
        - name: id
          type: string
          in: path
          required: true
      responses:
        '200':
          description: Product for given product id
          schema:
            $ref: '#/definitions/product'
        '404':
          description: Item not found
          schema:
            $ref: '#/definitions/error'
        '500':
          description: Server error
          schema:
            $ref: '#/definitions/error'
parameters:
  limitParam:
    name: $top
    in: query
    description: Limits the number of returned results
    required: false
    type: number
    format: int32
  offsetParam:
    name: $skip
    in: query
    description: Offset to the no of parameters to skip
    required: false
    type: number
    format: int32
definitions:
  error:
    type: object
    properties:
      code:
        type: string
        description: Error code
      message:
        type: string
        description: Error message
      description:
        type: string
        description: Detailed Error Description
  price:
```

| | ```
type: object
  properties:
    value:
      type: number
      format: double
      example: 33.99
      title: Product price
    currency:
      type: string
      title: Currency unit
      example: EUR
      enum:
        - USD
        - EUR
availability:
  type: object
  properties:
    availableAmount:
      type: integer
      example: 530
      title: Available product quantity
    warehouseId:
      type: string
      example: 890
      title: Warehouse Identifer
product:
  type: object
  properties:
    id:
      type: string
      description: Product Id
      example: 123
    name:
      type: string
      description: Product name
      example: KittyFood
    description:
      type: string
      description: Product description
      example: Tasty Kitty food
    price:
      $ref: '#/definitions/price'
    availability:
      $ref: '#/definitions/availability'
``` |

Code Generation

Step	Screenshot/ Sample
Generate code from your API to node.js **Go to** Generate Server – Node.js	

You can now generate your project. You can specify the Artifact and Artifact version.	**Project Metadata** Artifacts coordinates Artifact: user-application Artifact Version: 1.0.0 [Generate Project] [Cancel]
That will download a zip file containing your project, which can be unzipped to your local system.	nodejs-server.zip - WinRAR File Commands Tools Favorites Options Help Add Extract To Test View Delete Find Wizard Info VirusScan Comment nodejs-server.zip\user-application - ZIP archive, unpacked size 26.707 bytes Name — Size — Packed — Type — Modified — CRC32 .. — — — Local Disk api — — — File folder controllers — — — File folder .swagger-codeg... — 1.030 — 489 — SWAGGER-CODEG... — 25-05-18 09:47 — 5D47894C index.js — 1.412 — 655 — JavaScript File — 25-05-18 09:47 — 47DEFAA1 manifest.yml — 85 — 70 — Yaml Source File — 25-05-18 09:47 — 83C25742 mta.yaml — 283 — 173 — Yaml Source File — 25-05-18 09:47 — A8517BF9 package.json — 590 — 353 — JSON Source File — 25-05-18 09:47 — 3DD35AEC README.md — 2.081 — 706 — Markdown Source ... — 25-05-18 09:47 — 5BE4B31D
| Open Command Line tool and then navigate to root folder of the downloaded project.<br>Use command npm install to install all the dependent libraries<br><br>Use command npm start to run the application | ```
C:\Users\        \openSAP\user-application>npm install
up to date in 1.21s

C:\Users\        \openSAP\user-application>npm start

> products-catalog-api@1.0.0 prestart C:\Users\       \openSA
P\user-application
> npm install

up to date in 0.983s

> products-catalog-api@1.0.0 start C:\Users\        \openSAP\u
ser-application
> node index.js

Your server is listening on port 8080 (http://localhost:8080)
Swagger-ui is available on http://localhost:8080/docs
``` |
| Run the application | http://localhost:8080/docs |

FURTHER READS

- SAP Guidelines for REST API Harmonization
- OpenAPI Specification 2.0
- Accessing API Designer from SAP Cloud Platform Trial Account
- Documenting APIs with OpenAPI Specification in API Designer
- Video Tutorial for Creating and Documenting APIs in API Designer
- Blog on Develop and manage API first enterprise microservices with SAP Cloud Platform and API Management

Coding Samples

Any software coding or code lines/strings ("Code") provided in this documentation are only examples and are not intended for use in a production system environment. The Code is only intended to better explain and visualize the syntax and phrasing rules for certain SAP coding. SAP does not warrant the correctness or completeness of the Code provided herein and SAP shall not be liable for errors or damages cause by use of the Code, except where such damages were caused by SAP with intent or with gross negligence.

THE BEST RUN

®