Publishing to OpenAPI

You can convert CDS models to the OpenAPI Specification , a widely adopted API description standard.

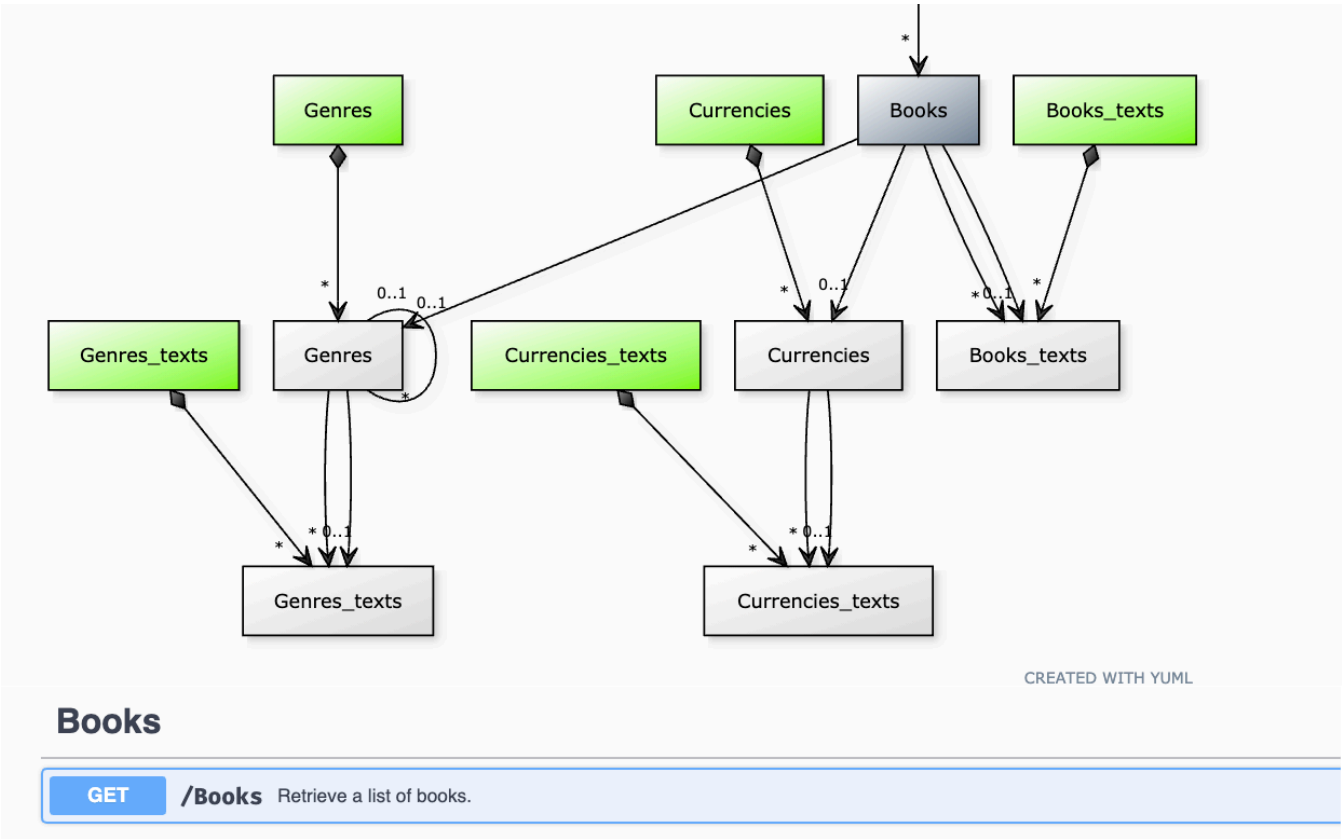
Table of Contents

Usage from CLI

For example, this is how you convert all services in `srv/` and store the API files in the `docs/` folder:

```sh
cds compile srv --service all -o docs --to openapi
```

With the `--openapi:diagram` parameter, you can also include a yuml entity-relationship diagram of the service entities in the Open API file.



The default value of the server URL is the service base path as declared in the CDS source.

If you have a **single server** and you want to set the server URL, use `--openapi:url <Server URL for Open API export>` option. Include the service path in the URL. For that, you can use the `${service-path}` variable.

If you want to configure **multiple servers**, you can use `--openapi:servers <JSON_Object_defining_servers>` which accepts stringified JSON of the server object. Here, you can pass multiple server objects by passing the stringified JSON objects as an array.

```sh
cds compile srv service.cds --to openapi --openapi:servers "\"'[{\\\"url\\\":
```

Note: `--openapi:url` is ignored when this option is specified.

Use the `--openapi:config-file <JSON_config_filepath>` option to provide configurations for all supported options in a configuration file. This file accepts a JSON format that incorporates all the OpenAPI compile options. Inline options take precedence over those defined in the configuration file.

```sh
cds compile srv service.cds --to openapi --openapi:config-file configFile.jso
```

Here is an example where `--openapi:config-file` option is used with other inline options:

```sh
cds compile srv service.cds --to openapi --openapi:config-file configFile.jso
```

In the above command, the `--openapi:diagram` and `--odata-version` inline options override the `--openapi:diagram` and `--odata-version` options in the *configFile.json* if they are also present there.

Swagger UI

Embedded in Node.js

In Node.js apps, the standard Swagger UI can be served with the help of the `cds-swagger-ui-express` package:

```sh
npm add --save-dev cds-swagger-ui-express
```

Swagger UI is then served at `$api-docs/...`. Just follow the *Open API preview* links on the index page:

Embedded in Java

Swagger UI is not available out of the box for CAP Java. However, check out this commit in our CAP Java sample application that demonstrates how to integrate a Swagger UI into your Spring Boot application.

Online Swagger Editor

Alternatively, you can use the online Swagger editor with the OpenAPI files produced with the CLI. In this case, you likely need to enable CORS because the `swagger.io` site needs to call `localhost`. You can use the `cors` middleware, for example.

Annotations

The OData to OpenAPI Mapping can be fine-tuned via annotations in the CSDL (`$metadata`) documents.

See Frequently Asked Questions for examples on how to use these annotations.

Core Annotations

Term	Annotation Target	OpenAPI field
`Computed`	Property	omit from Create and Update structures
`DefaultNamespace`	Schema	path templates for actions and functions without namespace prefix
`Description`	Action, ActionImport, Function, FunctionImport	`summary` of Operation Object
`Description`	EntitySet, Singleton	`description` of Tag Object
`Description`	EntityType	`title` of Request Body Object

Term	Annotation Target	OpenAPI field
Description	ComplexType, EntityType, EnumerationType, Parameter, Property, TypeDefinition	*title* of Schema Object
Description	Schema, EntityContainer	*info.title*
Example	Property	*example* of Schema Object
Immutable	Property	omit from Update structure
LongDescription	Action, ActionImport, Function, FunctionImport	*description* of Operation Object
LongDescription	Schema, EntityContainer	*info.description*
Permissions:Read	Property	omit read-only properties from Create and Update structures
SchemaVersion	Schema	*info.version*

Capabilities

Term	Annotation Target	OpenAPI field
CountRestrictions /Countable	EntitySet	*$count* system query option for *GET* operation
DeleteRestrictions /Deletable	EntitySet, Singleton	*DELETE* operation for deleting an existing entity
/Description	EntitySet, Singleton	*summary* of Operation Object
/LongDescription	EntitySet, Singleton	*description* of Operation Object
ExpandRestrictions /Expandable	EntitySet, Singleton	*$expand* system query option for *GET* operations
FilterRestrictions /Filterable	EntitySet	*$filter* system query option for *GET* operation

Term	Annotation Target	OpenAPI field
`/RequiredProperties`	EntitySet	required properties in `$filter` system query option for `GET` operation (parameter description only)
`/RequiresFilter`	EntitySet	`$filter` system query option for `GET` operation is `required`
`IndexableByKey`	EntitySet	`GET` , `PATCH` , and `DELETE` operations for a single entity within an entity set
`InsertRestrictions` `/Insertable`	EntitySet	`POST` operation for inserting a new entity
`/Description`	EntitySet	`summary` of Operation Object
`/LongDescription`	EntitySet	`description` of Operation Object
`KeyAsSegmentSupported`	EntityContainer	`paths` URL templates use key-as-segment style instead of parenthesis style
`NavigationRestrictions` `/RestrictedProperties`	EntitySet, Singleton	operations via a navigation path
`/DeleteRestrictions/...`	EntitySet, Singleton	`DELETE` operation for deleting a contained entity via a navigation path
`/FilterRestrictions/...`	EntitySet, Singleton	`$filter` system query option for reading related entities via a navigation path
`/InsertRestrictions/...`	EntitySet, Singleton	`POST` operation for inserting a new related entity via a navigation path
`/ReadByKeyRestrictions/...`	EntitySet, Singleton	`GET` operation for reading a contained entity by key via a navigation path
`/ReadRestrictions/...`	EntitySet, Singleton	`GET` operation for reading related entities via a navigation path
`/SearchRestrictions/...`	EntitySet, Singleton	`$search` system query option for reading related entities via a navigation path

Term	Annotation Target	OpenAPI field
/SelectSupport/...	EntitySet, Singleton	*$select* system query option for reading related entities via a navigation path
/SkipSupported	EntitySet, Singleton	*$skip* system query option for reading contained entities via a navigation path
/SortRestrictions/...	EntitySet, Singleton	*$orderby* system query option for reading related entities via a navigation path
/TopSupported	EntitySet, Singleton	*$top* system query option for reading contained entities via a navigation path
/UpdateRestrictions/...	EntitySet, Singleton	*PATCH* operation for modifying a contained entity via a navigation path
/Description	EntitySet	*summary* of Operation Object
/LongDescription	EntitySet	*description* of Operation Object
ReadRestrictions */Readable*	EntitySet, Singleton	*GET* operation for reading an entity set or singleton
/Description	EntitySet, Singleton	*summary* of Operation Object
/LongDescription	EntitySet, Singleton	*description* of Operation Object
ReadByKeyRestrictions */Readable*	EntitySet	*GET* operation for reading a single entity by key
SearchRestrictions */Searchable*	EntitySet	*$search* system query option for *GET* operation
SelectSupport */Supported*	EntitySet, Singleton	*$select* system query option for *GET* operation
SkipSupported	EntitySet	*$skip* system query option for *GET* operation
SortRestrictions */NonSortableProperties*	EntitySet	properties not listed in *$orderby* system query option for *GET* operation

Term	Annotation Target	OpenAPI field
/Sortable	EntitySet	*$orderby* system query option for *GET* operation
TopSupported	EntitySet	*$top* system query option for *GET* operation
UpdateRestrictions */Updatable*	EntitySet, Singleton	*PATCH* operation for modifying an existing entity
/Description	EntitySet, Singleton	*summary* of Operation Object
/LongDescription	EntitySet, Singleton	*description* of Operation Object
BatchSupport */Supported*	EntityContainer	*Batch* Support for the service

Validation

Term	Annotation Target	OpenAPI field
AllowedValues	Property	*enum* of Schema Object - list of allowed (string) values
Exclusive	Property	*exclusiveMinimum* / *exclusiveMaximum* of Schema Object
Maximum	Property	*maximum* of Schema Object
Minimum	Property	*minimum* of Schema Object
Pattern	Property	*pattern* of Schema Object

Authorization

Term	Annotation Target	OpenAPI field
Authorizations	EntityContainer	*securitySchemes* of Components Object/ *securityDefinitions* of Swagger Object
SecuritySchemes	EntityContainer	*security* of OpenAPI/Swagger Object

This is an example of a CDS service annotated with the annotations above:

```cds
annotate MyService with @(
  Authorization: {
    Authorizations: [
      { $Type : 'Authorization.Http', Name : 'Basic', Scheme : 'basic' },
      { $Type : 'Authorization.Http', Name : 'JWT',   Scheme : 'bearer', Bear
      { $Type : 'Authorization.OAuth2ClientCredentials', Name : 'OAuth2',
        Scopes     : [{
          Scope      : 'some_scope',
          Description: 'Scope description'
        }],
        RefreshUrl : 'https://some.host/oauth/token/refresh',
        TokenUrl   : 'https://some.host/oauth/token'
      },
    ],
    SecuritySchemes: [
      { Authorization : 'Basic' },
      { Authorization : 'JWT', RequiredScopes : [] },
      { Authorization : 'OAuth2' },
    ]
  }
);
```

↳ *See it in context.*

Common

Term	Annotation Target	OpenAPI field
Label	EntitySet, Singleton	*name* of Tag Object and entry in *tags* array of Operation Object

OpenAPI

Term	Annotation Target	OpenAPI field
externalDocs	EntityContainer	Links to external documentation that explain more about APIs are helpful to developers.
Extensions	EntityContainer	To add the sap defined (*x-sap*) specification extensions. This annotation can an be used in root, entity and in function/action level.

This is an example of a CDS service annotated with the annotations above:

```cds
annotate SampleService with @(
    OpenAPI:{
        externalDocs: {
            description: 'API Guide',
            url        : 'https://help.sap.com/docs/product/sample.html'
        },
        Extensions: {
        ![compliance-level]: 'sap:base:v1'
      }
    }
);
```

Frequently Asked Questions

Examples for typical questions on how to fine-tune the generated OpenAPI descriptions.

Suppress GET (list and by-key) on an entity set?

To suppress both types of GET requests to an entity set, annotate it with

```json
"@Capabilities.ReadRestrictions": {
    "Readable": false
}
```

Suppress GET (list) on an entity set?

To suppress only GET list requests to an entity set and still allow GET by-key, annotate it with

```json
"@Capabilities.ReadRestrictions": {
    "Readable": false,
    "ReadByKeyRestrictions": {
        "Readable": true
    }
}
```

Suppress GET (by-key) on an entity set?

To suppress only GET by-key requests to an entity set and still allow GET list, annotate it with

```json
"@Capabilities.ReadRestrictions": {
    "ReadByKeyRestrictions": {
        "Readable": false
    }
}
```

OData APIs AsyncAPI

Was this page helpful?