

Let's work out a deal

Contract First API Development with OpenAPI

API Specifications Conference 2021, Birgit Kratz

About me

Birgit Kratz

- Freelancing IT Consultant
- Java-Backend
- More than 20 years experience
- Co-Organizer of Softwerkskammer in Düsseldorf and Köln (Cologne)
- Email: mail@birgitkratz.de
- Twitter: @bikratz
- Github: <https://github.com/bkratz>



Agenda

What is an API

What is a REST(ful) API

Comparison of Code-First vs Contract-First API development approach

Tools supporting the Contract-First approach

Codegeneration

Demo

Experiences

What is an API

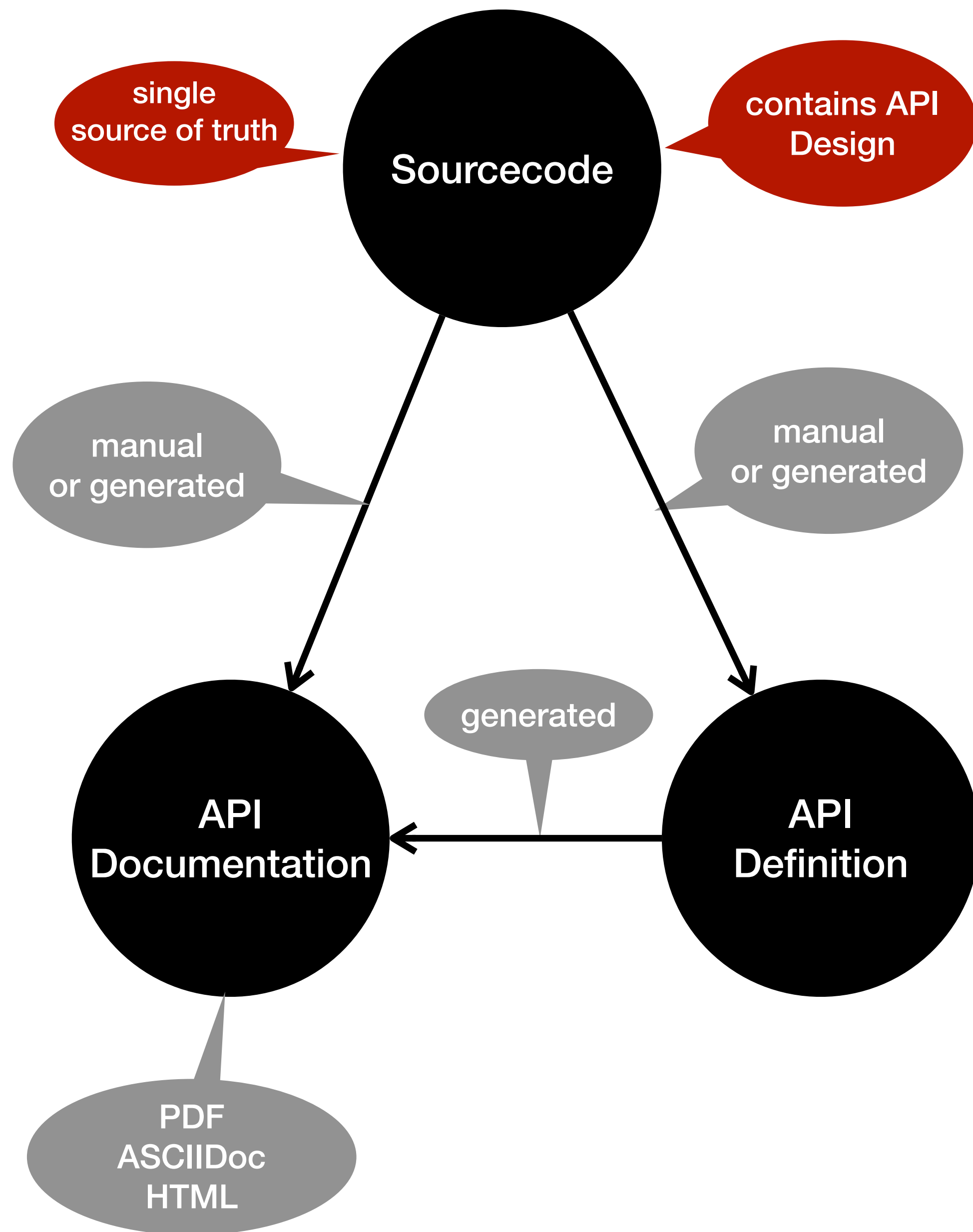
- **Application Programming Interface**
- “It is a type of software interface, offering a service to other pieces of software” (Wikipedia)
- Enables communication between computers or computer programs
- A document or standard that describes how to build such a connection or interface is called an *API specification*.
- Internal API, Partner/Customer API, Public/OpenSource API

What is a REST(ful) web API

- REST - Representational State Transfer
- RESTful web APIs are typically loosely based on [HTTP methods](#) to access [resources](#) via [URL-encoded](#) parameters and the use of [JSON](#) or [XML](#) to transmit data.
- Client-/Server communication, stateless, synchron
- Consistent access from a clients to resources of a server (a client can be a browser, a mobile app or another program “M2M”)
- Use of HTTP protocol (GET, POST, PUT, DELETE, ..., Authentication, Caching, Compression, Status Codes)
- HATEOAS - Hypermedia As The Engine Of Application State

How to develop a (RESTful) API

Code First



Advantages

If either already exists (legacy) code, it is a simple way to create an API definition and documentation. The resulting API definition can later turn into the single source of truth

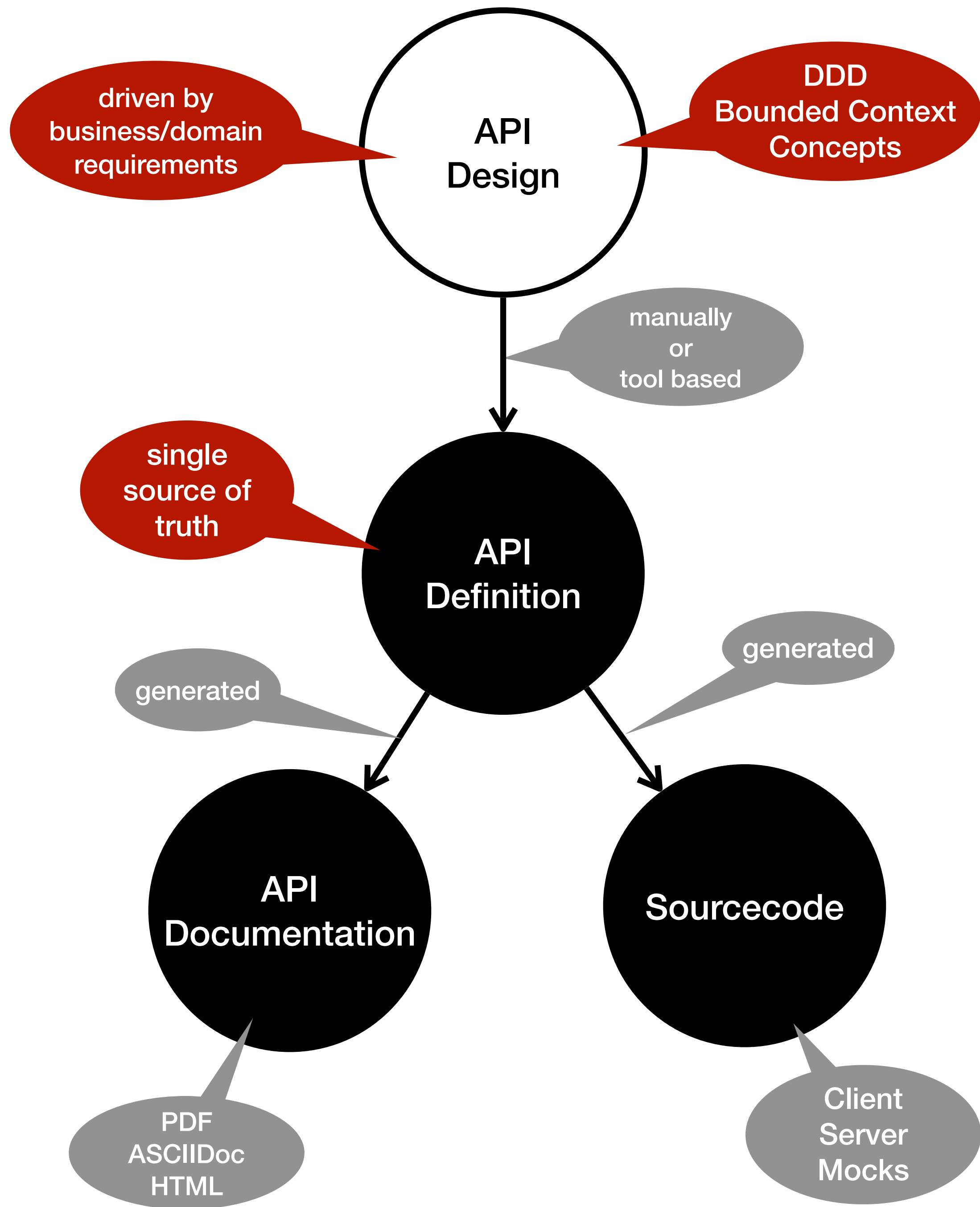
Disadvantages

API Definition has the tendency to become outdated if the code changes and it is not automatically created from the code

Client implementation only possible after server implementation has been finished and the API definition/documentation has been published

API Definition has the tendency to represent technical rather than domain aspects

API Definition First



Advantages

API Definition is driven by business/domain aspects and is single source of truth

Automatic generation source stubs and documentation

Server and client implementation can be done simultaneously and independent

Disadvantages

Tools that help generating source stubs sometimes do not cover all possibilities given by the API specification

Tools

OpenAPI Specification

<https://spec.openapis.org/oas/v3.0.3>

<https://swagger.io/specification>

<https://swagger.io/docs/specification>

Editors

Online Swagger Editor:

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

IntelliJ IDEA Plugin: OpenAPI (Swagger) Editor

<https://plugins.jetbrains.com/plugin/14837-openapi-swagger-editor>

Code Generator

openapi-generator-maven-plugin

<https://github.com/OpenAPITools/openapi-generator>

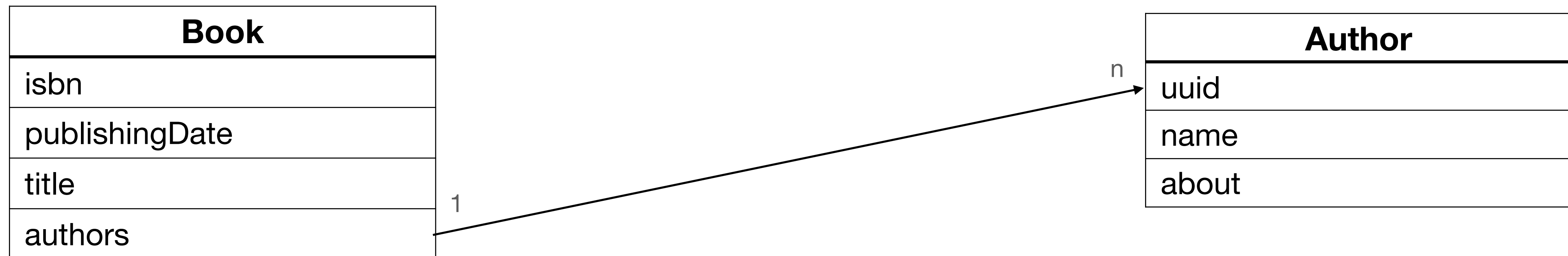
<https://github.com/OpenAPITools/openapi-generator/tree/master/modules/openapi-generator-maven-plugin>

<https://openapi-generator.tech/>

More Tools

<https://openapi.tools/>

Example project: Bookshelf



GET /books - get a list of all books
GET /books/{isbn} - get book details
POST /books - create a book
PUT /books/{isbn} - change a book
DELETE /books/{isbn} - delete a book

GET /books/author/{name}
get all books by author name

GET /authors/{uuid}/books
get all books for an specific author

GET /authors - get a list of all authors
GET /authors/{uuid} - get author details
POST /authors - create an author
PUT /authors/{uuid} - change an author
DELETE /authors/{isbn} -delete an author

Demo

Other topics

API Versioning

```
servers:  
  - url: /v1  
  - url: /v2
```



```
paths:  
  /books:  
    get:  
      operationId: fetchAllBooksV1  
      responses:  
        200:  
          description: list of books V1  
          content:  
            application/json;version1:  
              schema:  
                type: array  
                items:  
                  $ref: "#/components/schemas/Book"  
    get:  
      operationId: fetchAllBooksV2  
      responses:  
        200:  
          description: list of books v2  
          content:  
            application/json;version2:  
              schema:  
                type: array  
                items:  
                  $ref: "#/components/schemas/Book"
```



Security

Describing Security

```
components:  
  securitySchemas:  
    ApiKeyAuth:  
      type: apiKey  
      in: header  
      name: X-API-Key  
    OAuth2:  
      type: oauth2  
      flows: clientCredentials  
      tokenUrl: 'path/to/token/url'  
      scopes: {}
```

Applying Security

```
security:  
  - ApiKeyAuth: []  
  - OAuth2: []
```


Links != HATEOAS

```
responses:  
  '201':  
    description: Created  
    content:  
      application/json:  
        schema:  
          type: object  
          properties:  
            isbn:  
              type: string  
              format: "[1-9]{13}"  
    links:  
      GetBookByIsbn:  
        operationId: fetchByIsbn  
        parameters:  
          isbn: '$response.body#/isbn'
```

```
/books/isbn/{isbn}:  
  get:  
    operationId: fetchByIsbn  
    parameters:  
      - in: path  
        name: isbn  
        required: true  
        schema:  
          type: integer  
          format: int64  
    responses: ...
```

Thank you

Example code: https://github.com/bkrazt/contract_first_bookshelf

- Email: mail@birgitkrazt.de
- Twitter: @bkrazt
- Github: <https://github.com/bkrazt>