

DIE OPENAPI SPECIFICATION AKA SWAGGER

MEHR ALS NUR SCHNITTSTELLENBESCHREIBUNG

Dennis Kieselhorst | [IRIAN](#)

ÜBER MICH

- über 10 Jahre Erfahrung mit Java und verteilten Softwarearchitekturen
- früher Lead Entwickler und Architekt bei EWE TEL und T-Systems
- inzwischen Senior Consultant bei IRIAN
- Committer bei der Apache Software Foundation
- Mitglied in Organisationskomitees
 - Java User Group (JUG) Bremen-Oldenburg
 - Java Forum Nord



AGENDA

- Schnittstellen/ APIs allgemein
- Die Geburtsstunde von Swagger
- Gründung der Open API Initiative
- API-Entwicklung (Contract/ API vs. Code First)
- Alternativen zur OpenAPI Specification
- Praktische Anwendungsbeispiele
- Unterschiede zwischen Version 2.0 und 3.0.0
- Fazit

SCHNITTSTELLEN

- Application Programming Interfaces (APIs)
- früher häufig in der [Web Services Description Language \(WSDL\)](#) beschrieben
 - mit WSDL 2.0 technisch auch für REST-Schnittstellen möglich
 - aufgrund fehlender Ressourcenorientierung aber nur bedingt sinnvoll
- [Web Application Description Language \(WADL\)](#) ist ressourcenorientiert
 - gilt in der Praxis aufgrund der vorliegenden XML-Struktur aber als umständlich
 - wurde nie standardisiert

DIE GEBURTSTUNDE VON SWAGGER (1/3)

- 2010 durch Tony Tam ins Leben gerufen, da kein einfaches, unkompliziertes Beschreibungsformat für REST-APIs existierte
- Verwendung unterschiedlicher Technologien und Programmiersprachen in Kombination mit WSDL und WADL gestaltete sich schwierig
- Interface Definition Language (IDL) entwickelt und Open Source unter der Apache Lizenz veröffentlicht

DIE GEBURTSTUNDE VON SWAGGER (2/3)

Swagger bietet ein

- sprachneutrales und maschinenlesbares Format,
- definiert in JSON oder YAML,
- ermöglicht Contract/ API-First sowie Code-First Entwicklung und
- verfügt über einen Erweiterungsmechanismus.

DIE GEBURTSTUNDE VON SWAGGER (3/3)

Unterstützt wird dies durch grundlegendes Tooling in Form

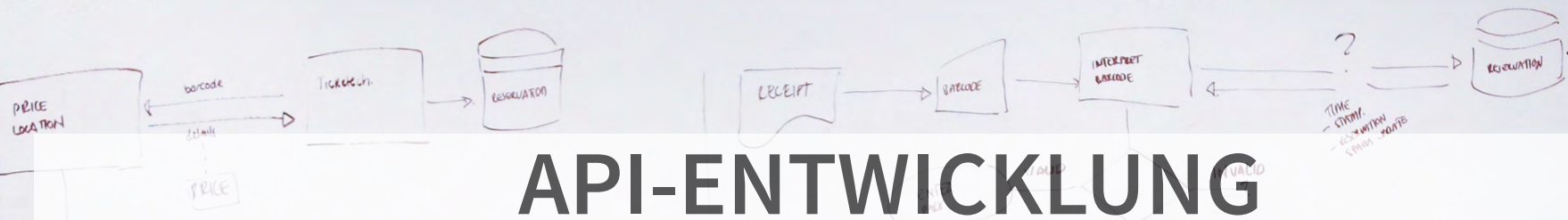
- einer Kernbibliothek (swagger - core),
- einer Oberfläche für Visualisierung und Testaufrufe (swagger - ui),
- eines Codegenerators (swagger - codegen) sowie
- sowie eines Editors (swagger - editor).

GRÜNDUNG DER OPEN API INITIATIVE (1/2)

- stetig wachsende Swagger Fan-Gemeinde
- Kauf von Swagger durch [SmartBear](#) Anfang 2015
- Gründung der [Open API Initiative \(OAI\)](#) unter dem Dach der Linux Foundation Ende 2015
 - herstellerneutral bleiben
 - unterschiedliche Interessen und Verbesserungsvorschläge besser handhaben

GRÜNDUNG DER OPEN API INITIATIVE (2/2)

- Im Zuge der Gründung wurde Swagger in OpenAPI Specification (OAS) umbenannt.
- Inzwischen zählt die Initiative **29 Mitglieder**, darunter Google, IBM und Microsoft.
- Weiterentwicklung der Spezifikation erfolgt auf GitHub
- Ende Juli 2017: Veröffentlichung der **Version 3.0.0**

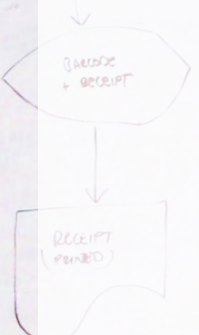


API-ENTWICKLUNG

Ausgangspunkt für die API-Entwicklung ist entweder die Schnittstellenbeschreibung (**Contract/ API-First**)

oder

der erstellte Programmcode (**Code First**).



CONTRACT/ API-FIRST

The image shows the Swagger Editor interface. On the left, a code editor displays the Swagger JSON definition for the Petstore API. On the right, the rendered API documentation is shown, including the title 'Swagger Petstore', version '1.0.0', base URL, and a list of endpoints for the 'pets' resource.

```

11  consumes:
12    - application/json
13  produces:
14    - application/json
15  paths:
16    /pets:
17      pet:
18        summary: List all pets
19        operationId: listPets
20        tags:
21          - pets
22        parameters:
23          - name: limit
24            in: query
25            description: How many items to return at one
26              time (max 100)
27            required: false
28            type: integer
29            format: int32
30        responses:
31          200:
32            description: An paged array of pets
33            headers:
34              x-next:
35                type: string
36                description: A link to the next page of
37                  responses
38            schema:
39              $ref: '#/definitions/Pets'
40          default:
41            description: unexpected error
42            schema:
43              $ref: '#/definitions/Error'
44      post:
45        summary: Create a pet
46        operationId: createPets
47        tags:
48          - pets
49        responses:
50          201:
51            description: Null response
  
```

Swagger Petstore
1.0.0
[Base url: petstore.swagger.io/v1]
MIT
Schemes: HTTP

pets

- GET /pets List all pets
- POST /pets Create a pet
- GET /pets/{petId} Info for a specific pet

Models

<http://editor.swagger.io>

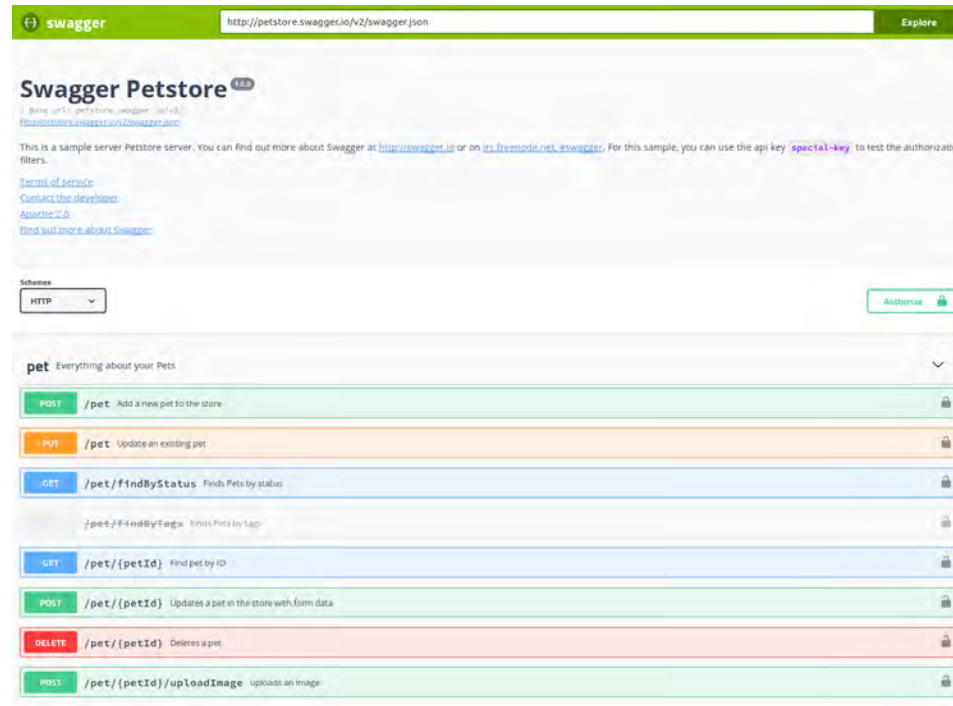
CONTRACT/ API-FIRST: VORTEILE

- Schnittstellenbeschreibung als vereinbarten *Vertrag*
 - bei unterschiedlichen Teams für Client- und Serverseite
 - häufig sogar in unterschiedlichen Unternehmen
- Client- und Servercode können mit [Swagger Codegen](#) direkt daraus generiert werden.
 - Code ist konsistent zur API
 - Kompilierungsfehler bei Brüchen (durch Continuous Integration System wie bspw. Jenkins automatisierbar)

CONTRACT/ API-FIRST: NACHTEILE

- Code wird ggf. etwas unübersichtlicher
- generelle Kritik an spezifikationsbasierter Codegenerierung ([Technology Radar](#))
 - besser *Tolerant Reader Pattern* anwenden
 - hängt von Umfang und Änderungshäufigkeit der API ab und muss im jeweiligen Projekt bewertet werden

VISUALISIERUNG DER API



Swagger UI

Alternativen: Swagger2Markup und ReDoc

CODE FIRST

- Eigenschaften werden im Code mit Annotations festgelegt
 - im **Beispiel** mit Java und JAX-RS
 - für .NET, PHP und weitere Sprachen gibt es aber ähnliche Möglichkeiten
- zur Laufzeit wird aus diesen die Swagger-Beschreibung generiert
 - abrufbar unter `/swagger.yaml` bzw. `/swagger.json` (plus ggf. konfiguriertem Pfad)
 - passende Ergänzungsmodul für **Jersey**, **RESTEasy** und **Mule**
 - Unterstützung in **Springfox** und **Apache CXF** (integrierte Swagger UI)

CODE FIRST: JAVA ANNOTATIONS

@Api

eine Ressource, ermöglicht Definitionen für alle darunterliegenden Operationen vorzunehmen, Daten von den JAX-RS Annotationen @Path, @Consumes und @Produces werden übernommen

@ApiOperation

eine einzelne Operation (Pfad und HTTP-Methode)

@ApiResponse

ein Rückgabewert nebst Fehlercode

@ApiParam

ermöglicht die Daten aus den JAX-RS Parameter Annotationen zu erweitern

@ApiModelProperty

Metadaten auf Klassenebene genutzt werden, die sich dann im Schema wiederfinden

@ApiModelPropertyProperty

konkrete Schemainhalte, zusätzlich werden auch JAXB und Bean Validation Annotationen verarbeitet

CODE FIRST: HELLO DEMO MIT CXF UND SPRING BOOT

```
git clone https://github.com/apache/cxf
cd distribution/src/main/release/samples/jax_rs/spring_boot
mvn spring-boot:run
x-www-browser http://localhost:8080/services/helloservice/\
api-docs?url=/services/helloservice/swagger.json
```

alternativ [Download der CXF Quellcode Distribution](#)

ALTERNATIVEN ZUR OPENAPI SPECIFICATION

- **API Blueprint**

- fokussiert sich stärker auf die menschliche Lesbarkeit und Einfachheit
- APIs werden in einer Markdown-ähnlichen Syntax beschrieben

- **RESTful API Modeling Language (RAML)**

- ist YAML-basiert und erlaubt komplexere Definitionen als Swagger 2.0
- Mulesoft als Ersteller ist inzwischen der Open API Initiative beigetreten

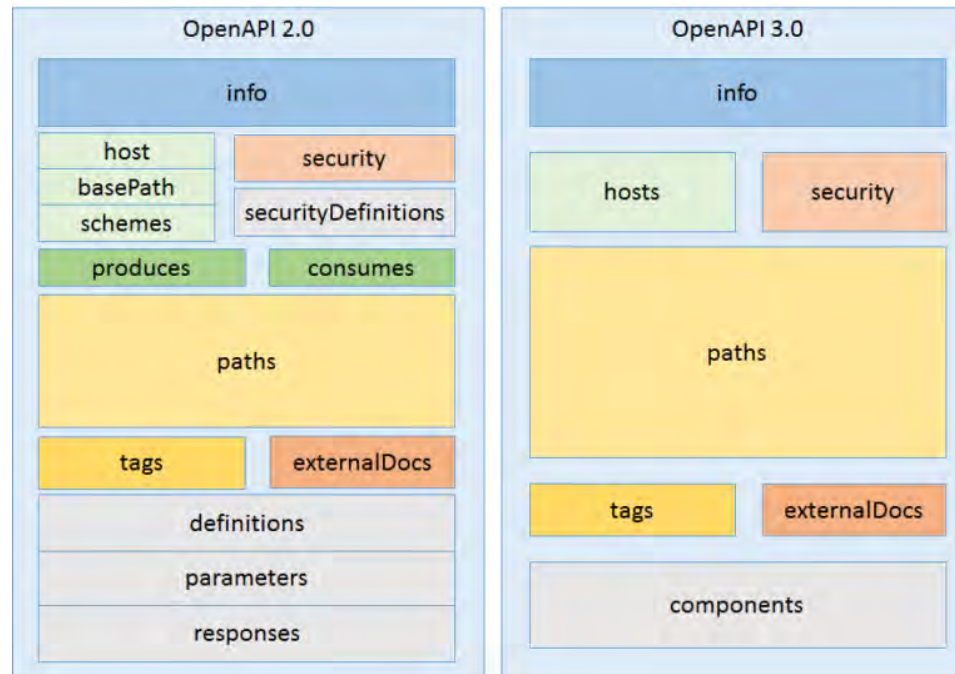
ANWENDUNGSBEISPIELE: ZALANDO

- einer der umsatzstärksten deutschen Online-Shops
- **Richtlinien** schreiben API-First Prinzip und Dokumentation nach OpenAPI Specification vor
- intern werden eine Vielzahl von APIs zwischen den Services autonomer Teams genutzt
- öffentliche **API zum Abruf von u.a. Artikeln und Bewertungen**

ANWENDUNGSBEISPIELE: LUFTHANSA

- größtes Luftverkehrsunternehmen Deutschlands
- stellt seit diesem Jahr mit der [Lufthansa Open API](#) eine Schnittstelle zur Verfügung mit der sich bspw. Flugpläne und Flugstatus auslesen lassen
- entwickelt vom Lufthansa Innovation Hub und kürzlich mit dem Digital Leader Award prämiert

UNTERSCHIEDE ZWISCHEN VERSION 2.0 UND 3.0.0



OpenAPI 2.0

info

host

basePath

schemes

security

securityDefinitions

produces

consumes

paths

tags

externalDocs

definitions

parameters

responses

OpenAPI 3.0

info

hosts

security

paths

tags

externalDocs

components

APICON

OpenAPI 3.0

APICON

Components

definitions

responses

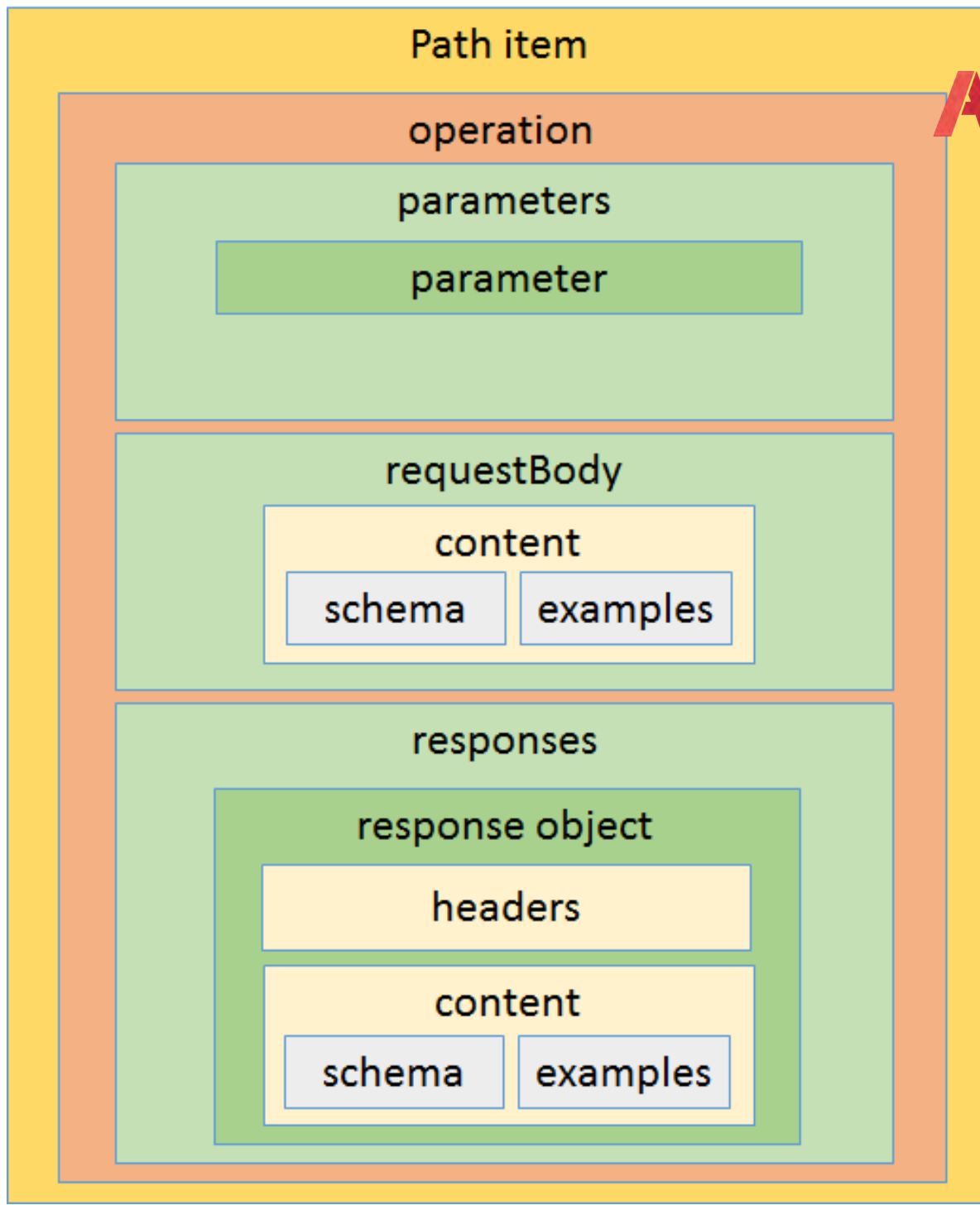
parameters

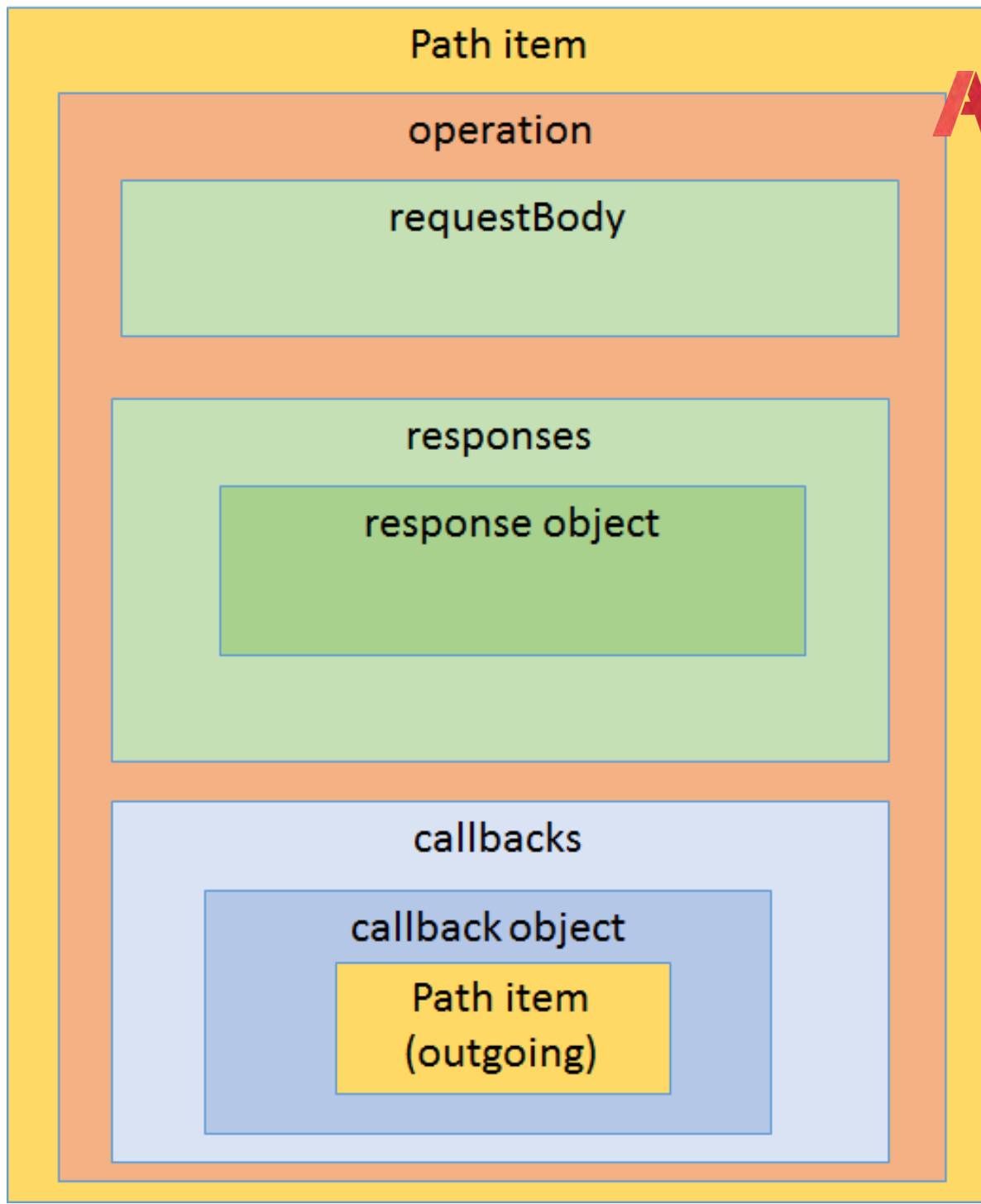
responseHeaders

securityDefinitions

callbacks

links





NICHT ENTHALTEN

- Alternative Schema wie [Google Protocol Buffers](#) sind in der Diskussion, wurden bislang aber noch nicht in die Specification integriert.
- Für asynchrone Anwendungsfälle wie MQTT und AMQP wurde daher basierend auf der OpenAPI Specification die [AsyncAPI Specification](#) ins Leben gerufen.



APICON

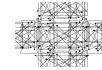
FAZIT

- Swagger 2.0 ist Quasi-Standard
 - hohe Verbreitung bei REST-basierten Anwendungen
 - umfangreiche Tool-Unterstützung
- OpenAPI Specification 3.0.0 sorgt für Ordnung
 - breites Gremium, in dem alle namhaften Hersteller vertreten sind
 - wird sich durchsetzen und Swagger 2.0 ablösen, sobald die Tool-Unterstützung angepasst ist

BILDQUELLEN



Pixabay geralt, CC0 Creative Commons



Pixabay geralt, CC0 Creative Commons



Pixabay congerdesign, CC0 Creative Commons



Pexels rawpixel.com, CC0 Creative Commons



Pexels startupstockphotos.com, CC0 Creative Commons



Pixabay Free-Photos, CC0 Creative Commons



Pexels Markus Spiske, CC0 Creative Commons



Pixabay MorganK, CC0 Creative Commons



Zalando Corporate Communications



Lufthansa Group Communications



Darrel Miller Open API Initiative



Darrel Miller Open API Initiative



Darrel Miller Open API Initiative



Darrel Miller Open API Initiative



Pixabay wilhei, CC0 Creative Commons



Huw Williams, Public Domain