BEDAG

REST-Schnittstellen

Dokumentation und Testing

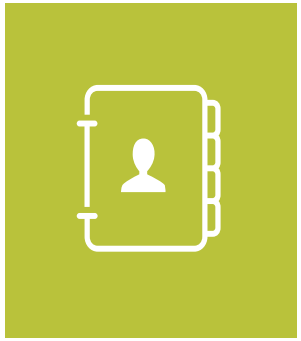
Adrian Moos (adrian.moos@bedag.ch)
Technology Advisor
Bedag Informatik AG

Inhalt

- Einleitung
- Dokumentation und Ad-Hoc-Testing
- Testautomation
- Fazit

BEDAG

Softwareentwicklung bei der Bedag



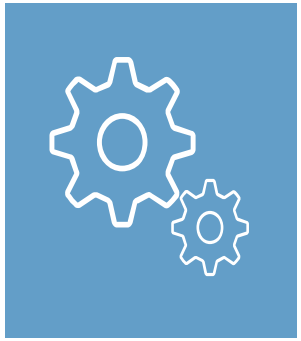
Sozialwesen
- Fallführung (KiSS)

Steuern & Personal
- Steuer-Bezug & Inkasso
- BE-Login
- Personalsystem

Register
- Einwohnerregister (Geres)
- Adressregister
- Unternehmensregister
- Stimm-/Wahlregister

Strassenverkehr
- Fahrzeuge
- Führerausweise
- Elekt. Versicherungsausweis

Raumsysteme
- Grundbuch (Capitastra)
- Amtliche Vermessung
- Gebäude- und Wohnungsregister
- Gebäudeschätzer



Diverses
- Atlassian Partner

BEDAG

Bedag Stack – Presentation

BEDAG

Bedag Stack – Java

HIBERNATE

Gibt es Standards?

	Operation Contract	Data Contract
SOAP / XML	WSDL	XML Schema
REST / JSON		

Gibt es Standards?

	Operation Contract	Data Contract
SOAP / XML	WSDL	XML Schema
REST / JSON	JSON-Hyperschema?	JSON-Schema?

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as “work in progress.”

This Internet-Draft will expire on August 3, 2013.

https://github.com/json-schema/json-schema/issues/167#issuecomment-113505647

http://json-schema.org/implementations.html

BEDAG

Was ist Swagger?

> “Swagger is a simple yet powerful representation of your RESTful API. With the largest ecosystem of API tooling on the planet, thousands of developers are supporting Swagger in almost every modern programming language and deployment environment. With a Swagger-enabled API, you get interactive documentation, client SDK generation and discoverability.”

- Open Source von SmartBear (Open API Initiative)
- Data & Operation Contract
  - in JSON oder YAML
  - Data Contract basiert auf JSON Schema
  - Operation Contract nicht

Swagger Tools

- Swagger-Editor
  - Web-GUI zum Bearbeiten einer Swagger-Spezifikation
  - kommerzielle Alternative: Restlet Studio
- Swagger-UI
  - Web-GUI zum Erforschen einer Swagger-Spezifikation
- Swagger-Codegen
  - Codegenerator für Client oder Server Stubs
  - Bindings für viele Programmiersprachen

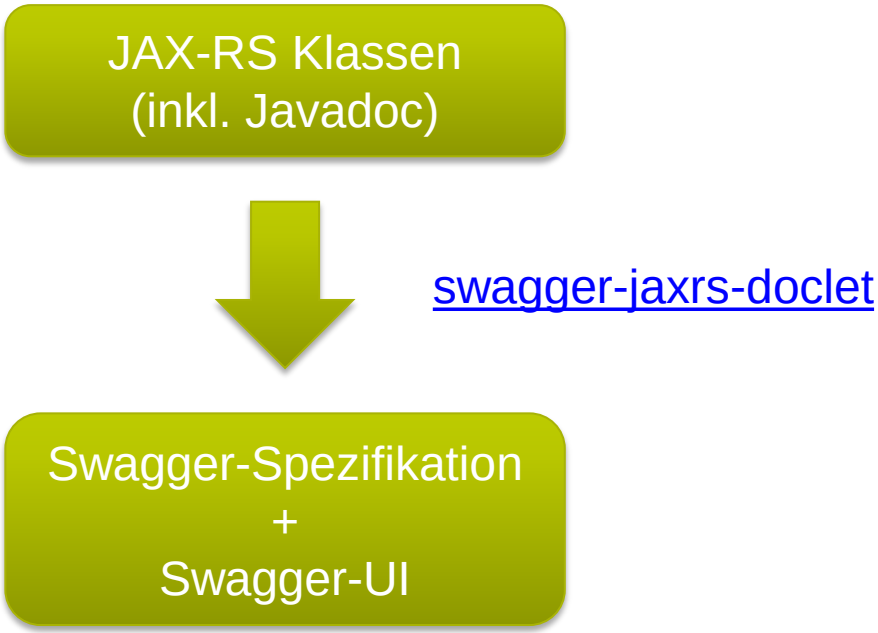
Inhalt

- Einleitung
- Dokumentation und Ad-Hoc-Testing
- Testautomation
- Fazit

Ziel

- discoverable
- übersichtlich
- vollständig
- korrekt (aktuell)
  - pflegeleicht
  - DRY

Methode



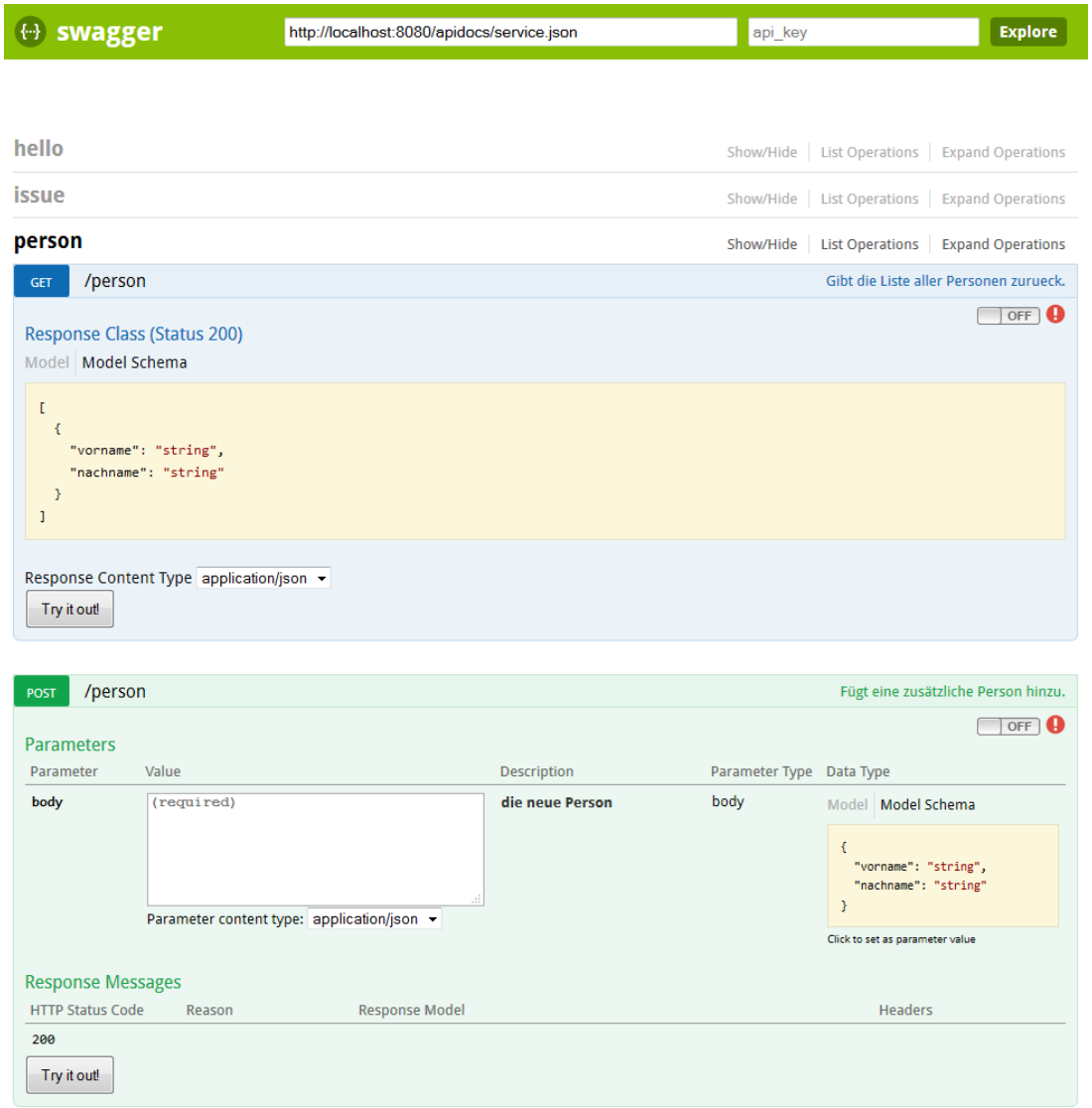
http://localhost:8080/apidocs

BEDAG

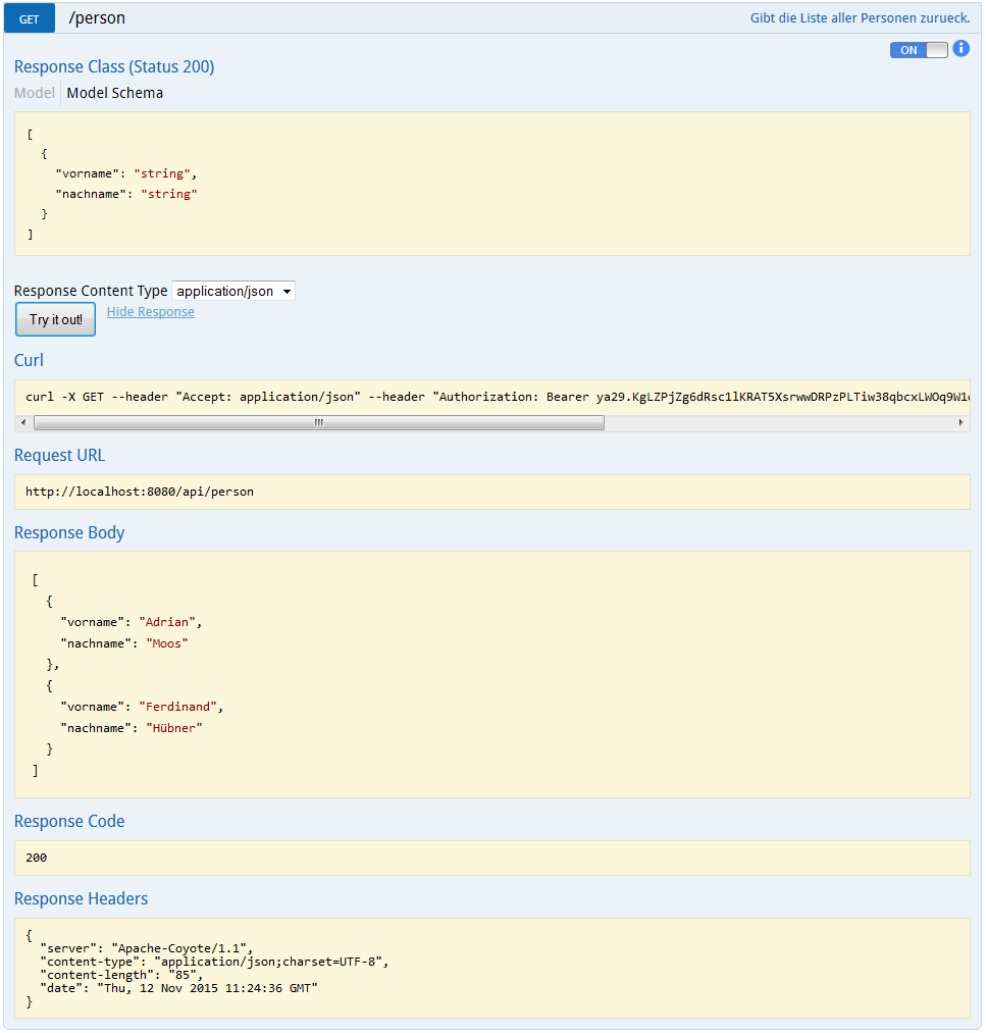
Demo

```
@Path("person")
public class PersonResource {

    private List<Person> persons = new ArrayList<>();

    /**
     * Gibt die Liste aller Personen zurueck.
     */
    @GET
    public List<Person> getAll() {
        return persons;
    }

    /**
     * Fügt eine zusätzliche Person hinzu.
     *
     * @param person die neue Person
     */
    @POST
    public void addPerson(Person person) {
        validate(person);
        persons.add(person);
    }
```

Demo


swagger
http://localhost:8080/apidocs/service.json
api_key
Explore
hello
Show/Hide
List Operations
Expand Operations
issue
Show/Hide
List Operations
Expand Operations
person
Show/Hide
List Operations
Expand Operations
GET
/person
Gibt die Liste aller Personen zurueck.
OFF
Response Class (Status 200)
Model
Model Schema
[
{
"vorname": "string",
"nachname": "string"
}
]
Response Content Type
application/json
Try it out!
POST
/person
Fügt eine zusätzliche Person hinzu.
OFF
Parameters
Parameter
Value
Description
Parameter Type
Data Type
body
(required)
die neue Person
body
Model
Model Schema
{
"vorname": "string",
"nachname": "string"
}
Parameter content type:
application/json
Click to set as parameter value
Response Messages
HTTP Status Code
Reason
Response Model
Headers
200
Try it out!

Demo

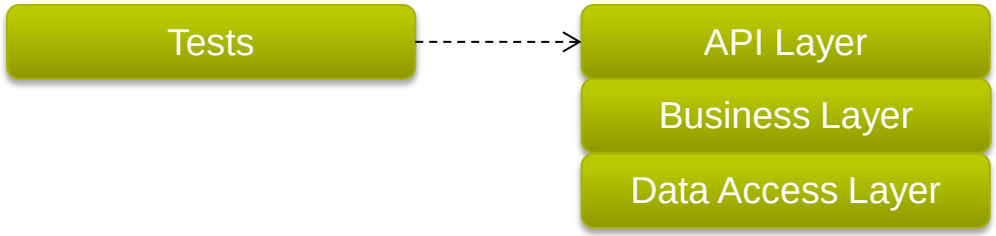

GET
/person
Gibt die Liste aller Personen zurueck.
ON
Response Class (Status 200)
Model
Model Schema
[
{
"vorname": "string",
"nachname": "string"
}
]
Response Content Type
application/json
Try it out!
Hide Response
Curl
curl -X GET --header "Accept: application/json" --header "Authorization: Bearer ya29.KgLZPjZg6dRsc1lKRAT5XsrwwDRPzPLTiw38qbcxLWOq9W1
Request URL
http://localhost:8080/api/person
Response Body
[
{
"vorname": "Adrian",
"nachname": "Moos"
},
{
"vorname": "Ferdinand",
"nachname": "Hübner"
}
]
Response Code
200
Response Headers
{
"server": "Apache-Coyote/1.1",
"content-type": "application/json;charset=UTF-8",
"content-length": "85",
"date": "Thu, 12 Nov 2015 11:24:36 GMT"
}

Inhalt

- Einleitung
- Dokumentation und Ad-Hoc-Testing
- Testautomation
- Fazit

Ziel

- Regressionstests
- integriert in CI-Build
- aussagekräftig
  - fachliche Abdeckung: komplexe Szenarien
  - technische Abdeckung: Full Stack



- wartbar
  - typsicher (damit die Entwicklungsumgebung helfen kann)
  - verständlich
  - prägnant

BEDAG

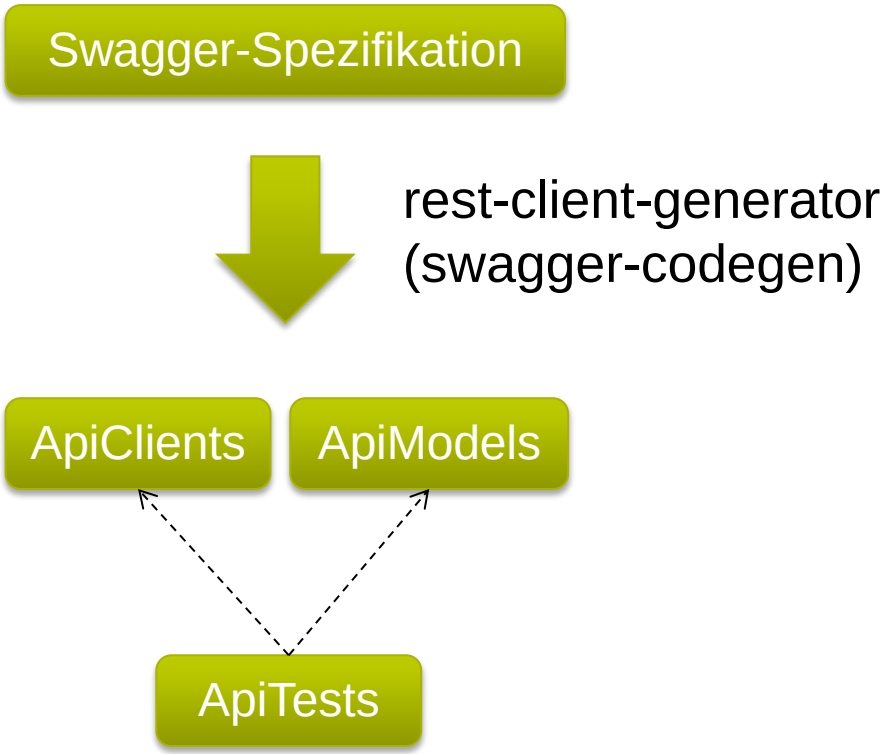
Gibt es schon sowas?

rest-assured ist sehr verbreitet:

```
get("/lotto").then()
    .assertThat().body("lotto.winners.winnerId", hasItems(23, 54));
```

- verständlich
- ziemlich prägnant
- nicht typsicher

Methode


Swagger-Spezifikation
rest-client-generator
(swagger-codegen)
ApiClients
ApiModels
ApiTests

Demo: generierte Model-Klasse

```
public class Person extends ApiModel {
    /**
     *
     **/
    @JsonProperty("vorname")
    public String vorname;

    /**
     *
     **/
    @JsonProperty("nachname")
    public String nachname;

}
```

Demo: generierte API-Klasse

```java
/**
 * {@link ApiClient} for the PersonResource.
 */
public class PersonApi extends ApiClient {

    /**
     * Gibt die Liste aller Personen zurueck.
     */
    public List<Person> getAll() {
        WebTarget target = baseTarget.path("/person");
        try {
            return target.request(JSON).get(new GenericType<List<Person>>() {});
        } catch (RuntimeException e) {
            throw decorate(e);
        }
    }

    /**
     * Fügt eine zusätzliche Person hinzu.
     * @param person die neue Person
     */
    public void addPerson(Person person) {
        WebTarget target = baseTarget.path("/person");
        try {
            target.request(JSON).post(Entity.entity(person, JSON), VOID);
        } catch (RuntimeException e) {
            throw decorate(e);
        }
    }
}
```

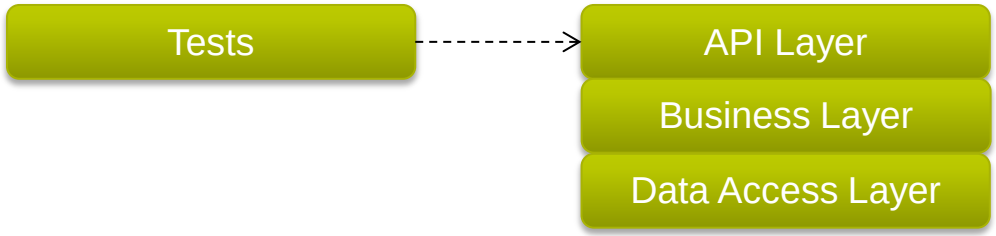
BEDAG

Demo: Ein typsicherer Test

```
public class PersonApiTest extends ExampleApiTest {

    private PersonApi api = new PersonApi();

    @Test
    public void addValidPerson() {
        Person person = new Person();
        person.vorname = "Peter";
        person.nachname = "Meier";

        api.addPerson(person);
        assertThat(api.getAll(), hasItem(person));
    }

    @Test
    public void addInvalidPerson() {
        Person person = new Person();
        person.vorname = "Peter";

        try {
            api.addPerson(person);
            fail();
        } catch (BadRequestException e) {
            assertThat(e.getMessage(), containsString("nachname may not be null"));
        }
    }
}
```

Ziel erreicht?

- Regressionstests
- integriert in CI-Build
- aussagekräftig
  - fachliche Abdeckung: komplexe Szenarien
  - technische Abdeckung: Full Stack



- wartbar
  - typsicher (damit die Entwicklungsumgebung helfen kann)
  - verständlich
  - prägnant

BEDAG

Umsetzung

- swagger-codegen reicht nicht
  - Generiert komplettes Maven-Projekt statt einen source folder
  - Generierte Klassen basierten auf Jersey 1.18
    - Mittlerweile undokumentiertes Feature für Jersey 2.6
  - Generierter Code sehr hässlich
  - Generierte Models implementieren kein equals()
- Deshalb:
  - Eigenes Maven-Plugin
  - Neues Language-Binding für Swagger-Codegen
    - Models des JavaClient-Bindings wiederverwendet
    - Komplett neue Templates
  - Code auf Github

Fazit

- Es gibt keinen Standard zur Beschreibung von REST Schnittstellen (aber gute proprietäre Lösungen)
- Mit Swagger können REST-Schnittstellen einfach dokumentiert, erforscht und interaktiv getestet werden
- Mit Swagger-Codegen können effizient aussagekräftige wartbare Regressionstests implementiert werden

Fragen?