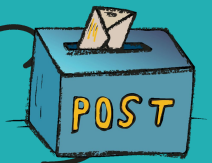
START

Agenda

1. Brief intro into RESTful API

2. Why do we test it

3. Rest Assured



4. Practice

What is a RESTful API

An API that:

- Uses HTTP/HTTPS connection and standard HTTP methods (e.g., GET, POST, PUT, PATCH and DELETE);
- Has a base URI
- Utilizes a media type that defines state transition data elements

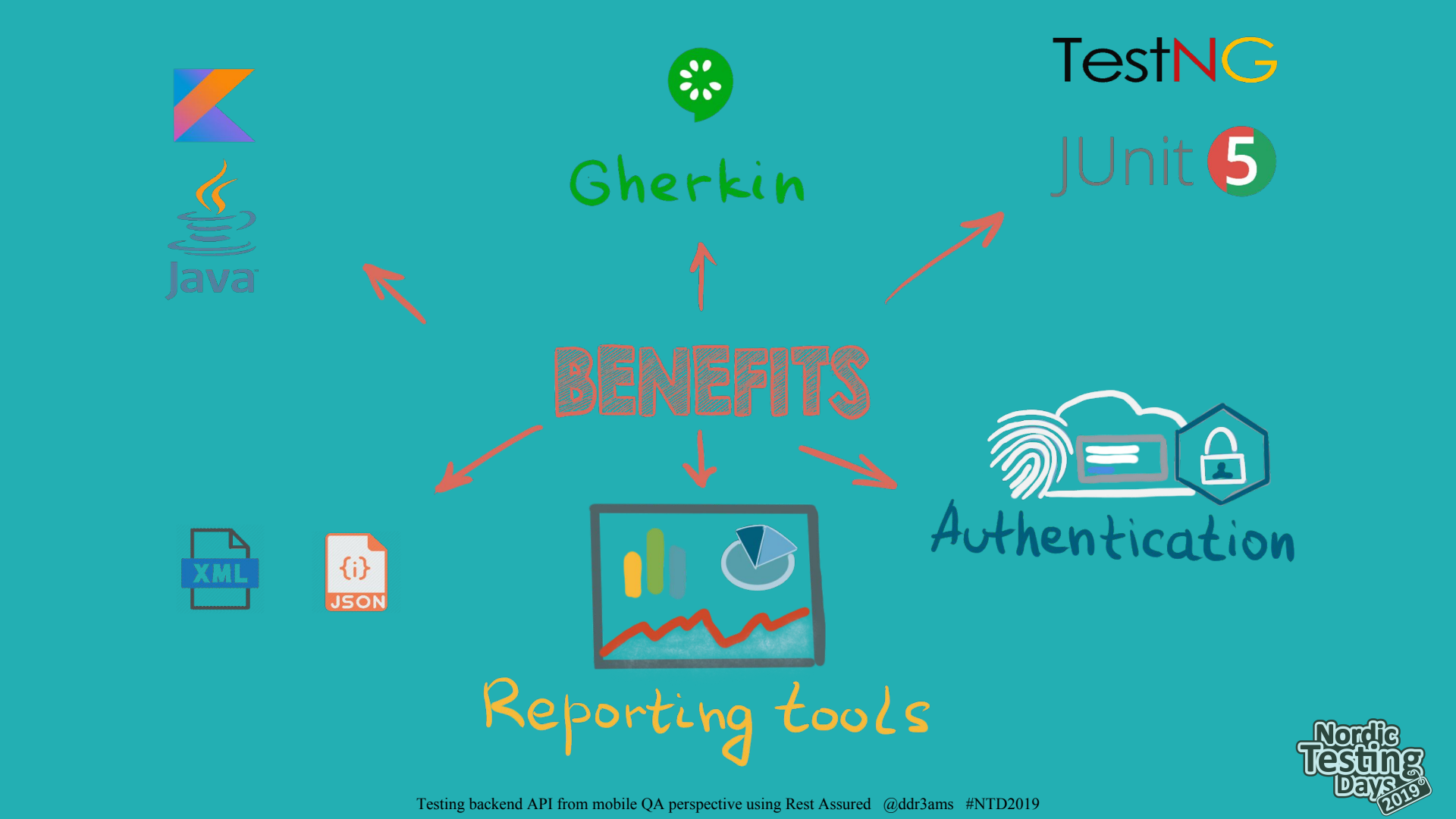
Why does a mobile QA test APIs

Lots of mobile applications have a backend:

- Make sure all endpoints work (status code, time)
- Make sure response structure is as expected
- Make sure data from backend is correct

What is REST-assured

- Java DSL for testing and validation of REST services.
- Created by Johan Haleby from Sweden
- First version was released on 24.12.2010
- Last version, ~~3.3.0 was released on 11.01.2019~~ 4.0.0 was released on 10.05.2019


TestNG
Gherkin
JUnit 5
Java
BENEFITS
Authentication
XML
JSON
Reporting tools

Rest Assured drawbacks

- Still have to learn Java/Kotlin
- Can be clumsier than using other languages => code reuse and patterns

Syntax

Assuming URI is "https://api.untappd.com/v4/", simple Rest Assured request looks like this:

- For a simple GET request:

  ```
  given().get("https://api.untappd.com/v4/path");
  ```

- For a GET request with header:

  ```
  given().header("key", "value").get("https://api.untappd.com/v4/path");
  ```

- For a GET request with a query param:

  ```
  given().param("key", "value").get("https://api.untappd.com/v4/path");
  ```

Syntax

- For a GET request with Form URL-Encoded params:

```
given().formParam("key", "value").get("https://api.untappd.com/v4/path");
```

- For a POST request with body:

```
given().body("{\"key1\": \"value1\", \"key2\": \"value2\"}")
        .post("https://api.untappd.com/v4/path");
```

Assertions

- In request code:

```
given().get("https://api.untappd.com/v4/path")
        .then.statusCode(200).statusLine("HTTP/1.1 200 OK");
```

- After getting response:

```
Response response = given().get("https://api.untappd.com/v4/path");
response.then().statusCode(200);
// same as: assert response.statusCode() == 200
response.then().statusLine("HTTP/1.1 200 OK");
```

Tests

Basic test example:

```
Response response =
        given()
                .formParam("user_name", "Someuser")
                .formParam("user_password", "SomePass")
                .post("https://api.untappd.com/v4/xauth");
response.then().statusCode(200);
assert response.body().path("response.access_token") != null;
```

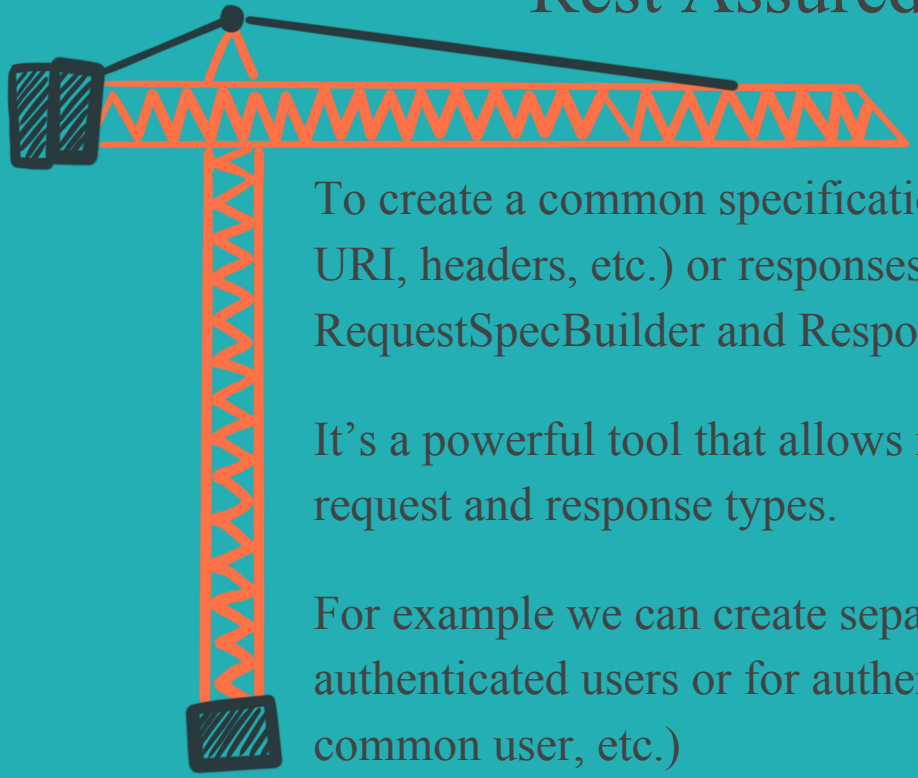
Authentication

Basic: given().auth().basic("username", "password");

Oauth2: given().auth().oauth2(accessToken);

Custom header: given().header("Authorization", "Basic " + token);

Rest Assured Spec Builders



To create a common specification for several requests(e.g. having the same URI, headers, etc.) or responses (same status code, headers, etc.) we can use RequestSpecBuilder and ResponseSpecBuilder.

It’s a powerful tool that allows reduction of duplicated code and structuring of request and response types.

For example we can create separate request specs for non authenticated and authenticated users or for authenticated users with different roles (admin, common user, etc.)

RequestSpecificationBuilder

Create a common request builder:

```
RequestSpecificationBuilder requestSpecBuilder = new RequestSpecificationBuilder();
requestSpecificationBuilder.setBaseUri("https://api.untappd.com/v4/");
requestSpecificationBuilder.setContentType("application/json");
RequestSpecification requestSpec = requestSpecBuilder.build();
```

Create an authenticated request builder:

```
RequestSpecificationBuilder authenticatedRequestSpecBuilder = new RequestSpecificationBuilder();
requestSpecificationBuilder.setBaseUri("https://api.untappd.com/v4/");
requestSpecificationBuilder.setContentType("application/json");
requestSpecificationBuilder.addHeader("Authorization", "Bearer " + token);
RequestSpecification authenticatedRequestSpec = requestSpecBuilder.build();
```

ResponseSpecificationBuilder

Create a response spec for all responses that should have a “200 OK” status:

```
ResponseSpecBuilder responseSpecBuilder = new ResponseSpecBuilder();
responseSpecBuilder.expectStatusCode(200);
ResponseSpecification response200Spec = responseSpecBuilder.build();
```

Use specifications

- **In request:**

```
given().spec(authenticatedRequestSpec).get("/auth/")
```

- **In response:**

```
given().get("https://api.untappd.com/v4/path").then().spec(response200Spec)
```

- **Both:**

```
given().spec(authenticatedRequestSpec).get("/auth/")
        .then().spec(response200Spec)
```

Response validation (JSON)

Our tests should check that response structure is correct, via:

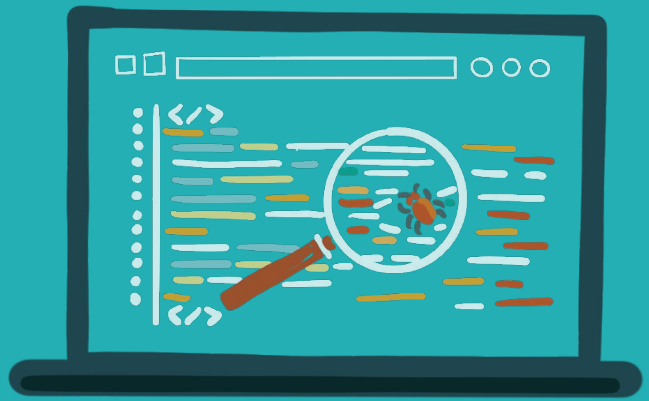
- **JSON Schema:**

```
given().get("https://api.untappd.com/v4/path").then().assertThat()
        .body(matchesJsonSchemaInClasspath("Authenticate.json"));
```

- **Serializing response body as Java object:**

```
Response response = given().get("https://api.untappd.com/v4/path");
AuthResponseBody responseBody = response.as(AuthResponseBody.class);
```

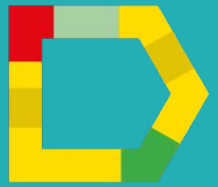
Debugging



- **Proxy:**
  - RestAssured.proxy("localhost", 8200);
  - requestSpecificationBuilder
    .setProxy("localhost", 8200);

- **Printing request and responses:**

```
requestSpecificationBuilder.addFilter(new RequestLoggingFilter());
requestSpecificationBuilder.addFilter(new ResponseLoggingFilter());
```



Allure reports

Allure - an open-source framework designed to create test execution reports that are clear to everyone in the team.

Pros:

- Open sourced
- Easy to integrate with Rest Assured
- Supports attachments: screenshots, videos, logs, etc.
- A lot of customization: named steps, tests description, etc.
- Shows trends and history
- Marks flaky tests
- Works with all popular CI tools

Default report

Test Summary

44	4	0	53.592s
tests	failures	ignored	duration

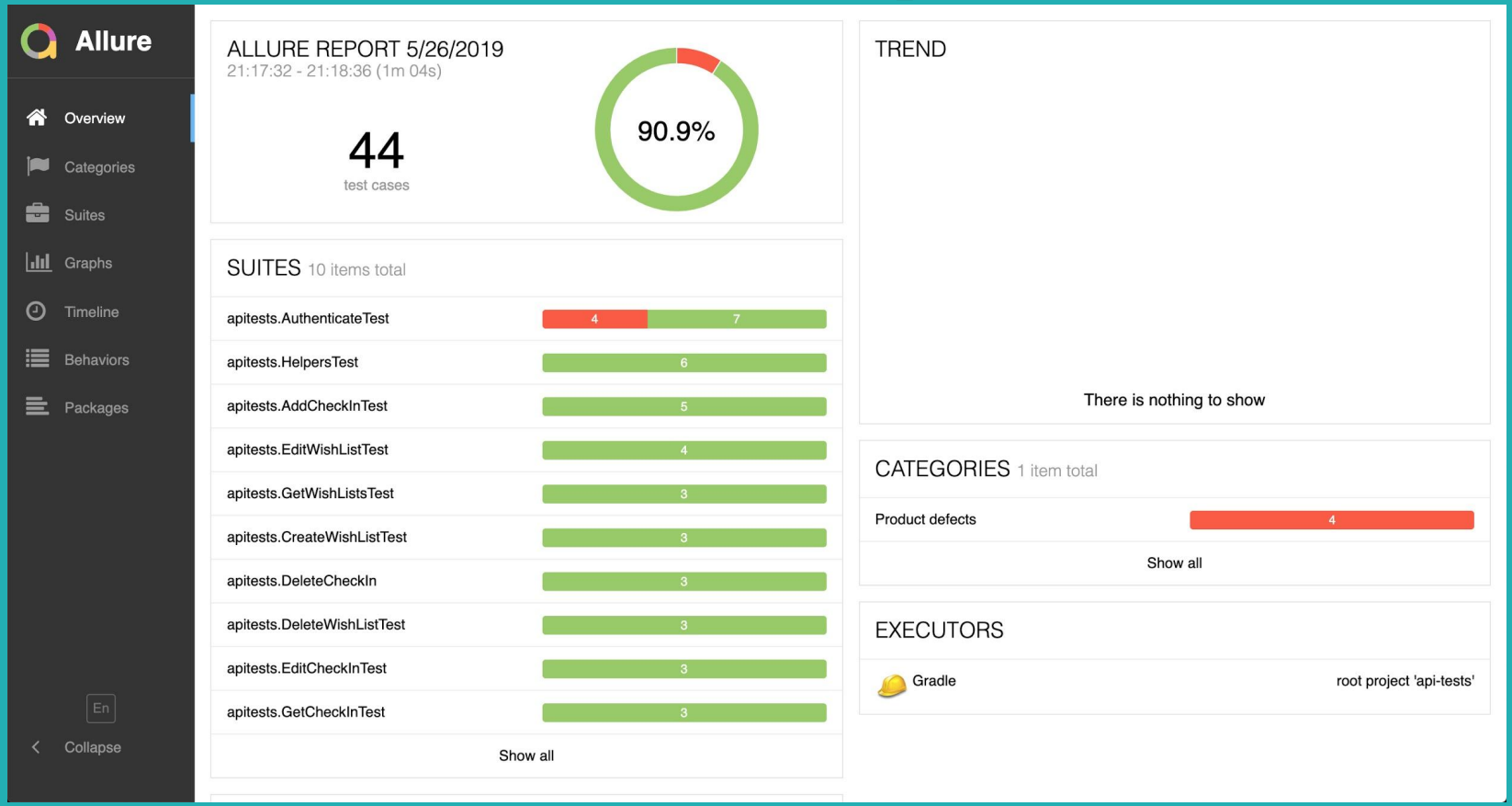
90% successful

Failed tests | Packages | Classes

AuthenticateTest. authJsonBodyTest()
AuthenticateTest. 2 authenticateNegativeWithParametrizedDataTest: No password. Username: username; Password: . Expected status code: 302. Expected error type: invalid_param
AuthenticateTest. authenticateWithValidCredsTestAssertResponseBodyByJsonSchema()
AuthenticateTest. authenticateWithValidCredsTestAssertTokenByJsonPath()

Generated by Gradle 5.4.1 at May 26, 2019, 9:18:37 PM

Allure report



Demo

1. Code walk through
2. Pre-recorded demo of tests execution and reporting using Rest Assured and Allure

Useful links

http://rest-assured.io

https://github.com/rest-assured/rest-assured

https://github.com/rest-assured/rest-assured/wiki/usage

https://github.com/allure-framework/allure-java

https://testautomationu.applitools.com/automating-your-api-tests-with-rest-assured/

https://issart.com/blog/practical-guide-for-making-tests-execution-result-reports-more-comprehensible

https://medium.com/gradeup/rest-api-testing-using-rest-assured-56a6cf772ca3

https://untappd.com/api/docs

https://jsonpath.com/

https://github.com/onikiforov/rest-assured-untappd