

# From zero to API hero with Java and REST-Assured



TechChamps



---

TUTORIAL





TechChamps

# WHOA!

- Arjan Assink
- Enschede The Netherlands
- 10+ years in test automation
- Currently working at TechChamps



# Topics



TechChamps

- APIs
- REST-Assured (incl. Key features)
- Apply DRY (Don't repeat yourself)
- Use Helper Class
- Authorization
- Request chaining
- DTOs (Data Transfer Objects)
- Property files



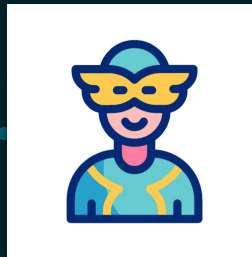
You will have readable and maintainable code



TechChamps

# Goal

- Start as zero
- Learn:
  - the basics with verbose code
  - to optimize code for readability and maintainability
  - to parameterize code for scalability
  - to test E2E flows
  - to use DTOs
- End as API hero



# What are APIs



TechChamps

- Application Programming Interface
- APIs are like digital contracts that allow different software applications to communicate and work together.
- APIs are everywhere

# What are APIs



TechChamps

Different types of APIs:

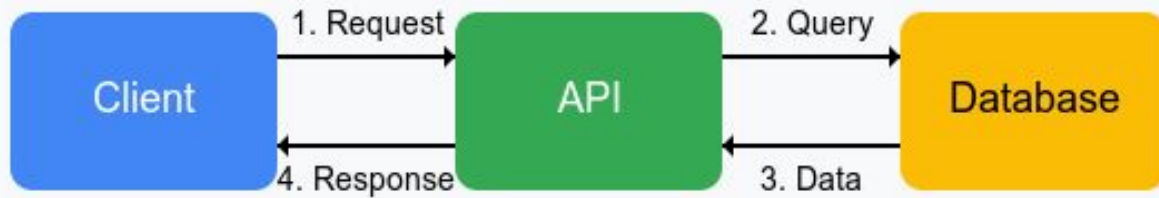
- REST (Representational State Transfer)
  - Resource-based with standard HTTP methods (GET, POST, PUT, DELETE)
  - Stateless and cacheable
  - Uses standard media types (JSON, XML)
  - Most common type for web services
- SOAP (Simple Object Access Protocol)
- GraphQL (Query language for APIs)
- And more

# API example



TechChamps

## Simple API Request Flow



# Example of a request



TechChamps

## Example of an API Request

METHOD

**POST** /api/users HTTP/1.1

HEADERS

### Headers

```
Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IHR5cGU6IjY...  
Content-Type: application/json  
Accept: application/json  
Content-Length: 152  
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
```

BODY

### Request Body (Payload)

```
{  
  "firstName": "Jane",  
  "lastName": "Smith",  
  "email": "jane.smith@example.com",  
  "password": "securePassword123",  
  "role": "editor"  
}
```



# Example of a response



TechChamps

## Anatomy of an API Response

STATUS

HTTP/1.1 200 OK

HEADERS

### Response Headers

Content-Type: application/json  
Cache-Control: max-age=3600  
X-RateLimit-Remaining: 49  
Date: Thu, 13 Mar 2025 15:23:45 GMT

BODY

### Response Body

```
{  
  "id": 42,  
  "name": "John Doe",  
  "email": "john.doe@example.com",  
  "role": "admin"  
}
```



TechChamps

# Status Codes

## 1xx - Informational

100 Continue, 101 Switching Protocols

## 2xx - Success

200 OK, 201 Created, 204 No Content

## 3xx - Redirection

301 Moved Permanently, 302 Found, 304 Not Modified

## 4xx - Client Errors

400 Bad Request, 401 Unauthorized, 403 Forbidden, 404 Not Found

## 5xx - Server Errors

500 Internal Server Error, 502 Bad Gateway, 503 Service Unavailable

HTTP Status Codes Overview



TechChamps

# Path parameters and Query parameters

## Example 1: Path Parameter

https://api.example.com

/users

/42

Path Parameter (userId = 42)

## Example 2: Query Parameters

https://api.example.com

/products

?

category=electronics  
price\_min=100price\_max=500

Query Parameters (filtering and sorting options)

GET requests showing different parameter types in API URLs



TechChamps

# Key benefits of API testing

**Fast execution**

**Easier to test  
backend logic**

**More reliable**

**Most business  
logic is handled by  
APIs**

**Earlier detection of  
issues**

**Easy to integrate in  
CI/CD pipeline**

**Independent of UI  
framework**



TechChamps

# What is REST-Assured

**Java library for  
testing REST APIs**

**Simple  
Given/When/Then  
syntax**

**Extensive  
capabilities for  
composing HTTP  
requests**

**Easy response  
validation**

REST-Assured simplifies the process of testing RESTful APIs by providing a readable and fluent syntax for writing tests.



TechChamps

# Why REST-Assured

**Documentation with good examples (Usage Guide)**

**Easy integration with e.g. Wiremock, Cucumber, JUnit**

**Developers within the team onboard with Java**

**Frequently used tool**

**Open source**



TechChamps

# Key features of REST-Assured

**Simple Syntax for  
HTTP Requests**

**Request and  
Response logging**

**Support for JSON  
and XML**

**Parameterization**

**Integration with  
testing  
frameworks**

**Built-in Assertions**

**Support for file  
upload and  
downloads**

**Custom header  
and request  
specification**

**Support for  
authentication**

**Custom  
(de)serialization**



TechChamps

# Basic example of REST-Assured

## RestAssured API Test

```
Given().baseUrl("https://api.example.com")
```

```
When().get("/users/1")
```

```
Then().statusCode(200)
```

```
.body("name", equalTo("John Doe"))
```

RestAssured test for validating an API response





TechChamps

# Request specification example

## RequestSpecification

- baseURI / basePath / port
- headers / contentType / cookies
- parameters / authentication

### given()

- spec(requestSpecification)
- additional headers
- request body

### when()

- HTTP method (get, post)
- Endpoint path
- Executes the request

### then()

- statusCode verification
- body assertions
- header validations

# API used for this tutorial



TechChamps

**Spring Boot**

**OpenApi  
(Swagger)**

**Maven for  
dependency  
management**

**Authentication  
with JSON Web  
Token**

**Roles (admin,user)  
for authorisation**

**H2 in memory  
database**

## Testing:

**REST-Assured**

**JUnit**

**Maven for  
dependency  
management**



TechChamps

**Let's look a basic sign in test**



TechChamps

# Summary

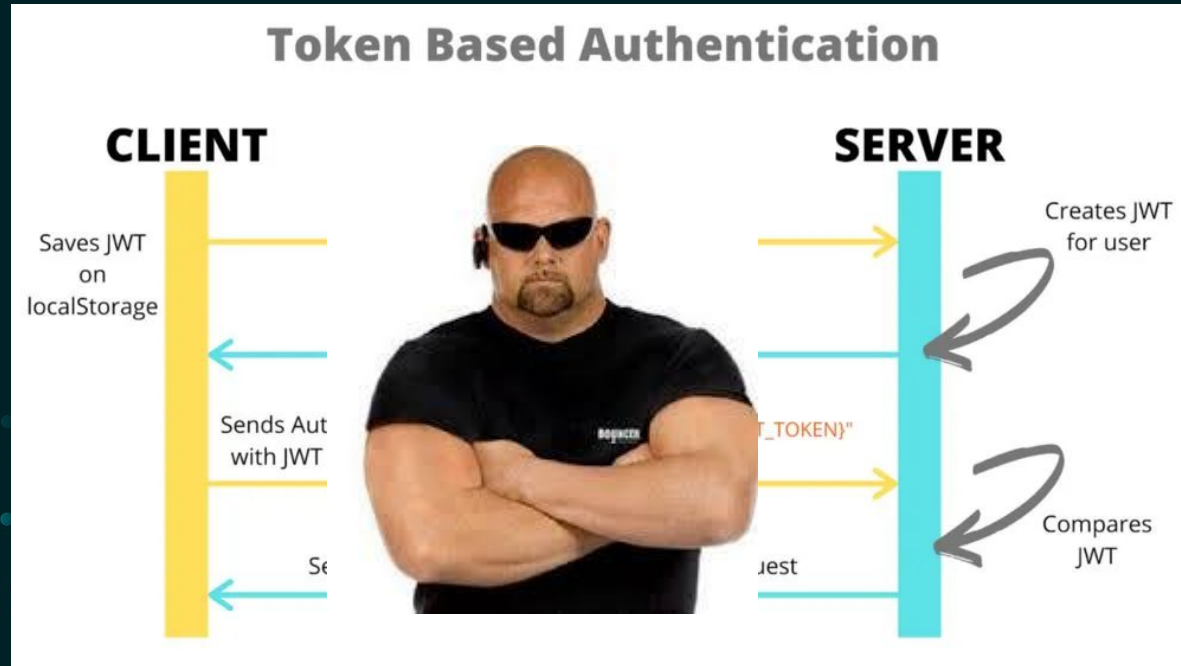
- Applied DRY (Don't Repeat Yourself)
- Introduced a Helper class with a RequestSpecBuilder
- This helper class will be enhanced later on

# Authentication



TechChamps

How does JWT (JSON Web Token) authentication work?





TechChamps

**Let's look at an example  
and how to improve that**



TechChamps

# Summary

- Used authentication
- Example of request chaining
- Put the authentication part in the Helper class to make the tests more readable and maintainable



TechChamps

# What is request chaining

Request chaining is a technique where multiple API requests are executed in sequence, with each subsequent request depending on data received from previous responses.



# Why is request chaining useful



TechChamps

Allows you to test  
E2E workflows

Dynamic testing

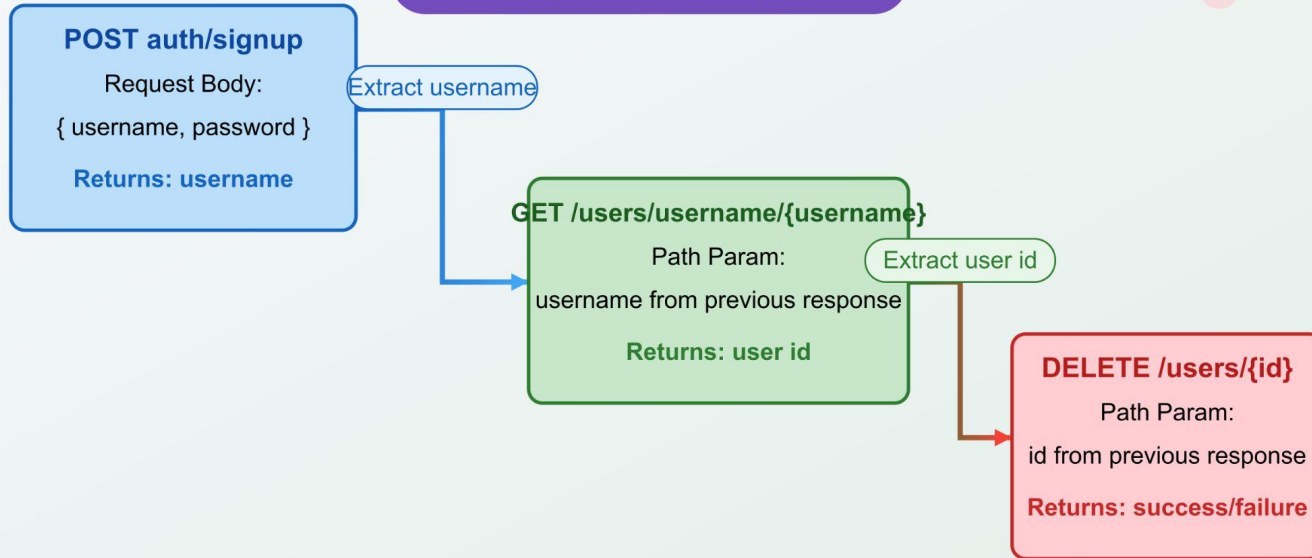
Data validation

# Let's look at an example



TechChamps

## API Request Chaining Flow





TechChamps

**Let's implement it in our  
code**



TechChamps

# DTO (Data Transfer Object)

DTO (Data Transfer Object) is a design pattern used in APIs to transfer data between layers, systems, or modules. In the context of APIs, DTOs are objects that encapsulate data and serve as a structured way to pass information between the client and the server.

# Why Use DTOs in APIs?



TechChamps

**Adaptability to  
client needs**

**Reduce overhead**

**Improved security**

**Separation of  
concerns**



TechChamps

# Why Use DTOs in REST-Assured

Using DTOs (Data Transfer Objects) in REST-Assured instead of raw JSON has several advantages:

**Type safety**

**Increase readability**

**Increase maintainability**

**Easier to debug**

**Reusability**

**Autocompletion**



TechChamps

**Let's look at our API and  
use DTOs in our test**

# Don't copy/create DTOs but auto generate them



TechChamps

Code duplication

Maintenance  
overhead

Consistency





TechChamps

# Let's autogenerate our DTOs

- **Springdoc-openapi-maven-plugin**
- **Openapi-generator-maven-plugin**

Let's have a look



TechChamps

# Why use Property files

**Separation of  
concerns**

**Flexibility**

**Environment -  
specific  
configurations**

**Security**

**Ease of  
Deployment**

**Consistency**

**Centralized  
Configuration  
Management**



TechChamps

**Let's implement it in the code**

# Tips



TechChamps

- Integrated in CI/CD from the beginning
- Let developers review your code
- API testing is more than checking status codes and responses. Test E2E flows
- Start simple and then improve it
- Use generated DTOs
- Keep in mind DRY
- Don't make it too abstract. keep it readable for other testers

# Questions?



TechChamps

- Repository
  - Main
  - Tutorial

