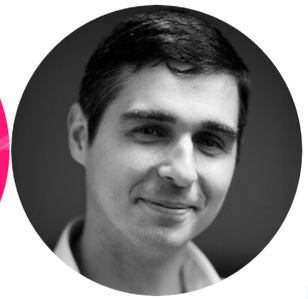
REST Assured Fundamentals

Introducing REST API Testing



Andrejs Doronins
Software Developer in Test

REST Assured Fundamentals

Version Check

Version Check

This course was created by using:

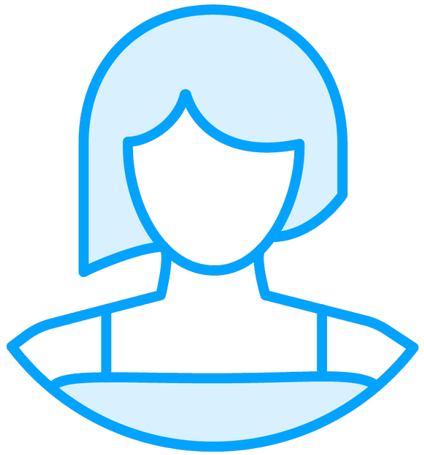
- REST Assured 5.3
- Java 17

Version Check

This course is 100% applicable to:

- REST Assured 5.x
- Java 17

Why learn REST Assured?



Why should I learn it?

To gain API testing skills

Is there demand?

Yes!

Speed and efficiency matter!

Which tool should we use?

GUI-based

Code-based

Advantages of GUI Tools

Lower learning curve

Works out-of-the-box

Suitable for both simple and complex APIs

But I also need it do X!

GUI Tool:

has 100+ features,
but not what you need

Code-based Solutions

REST Assured

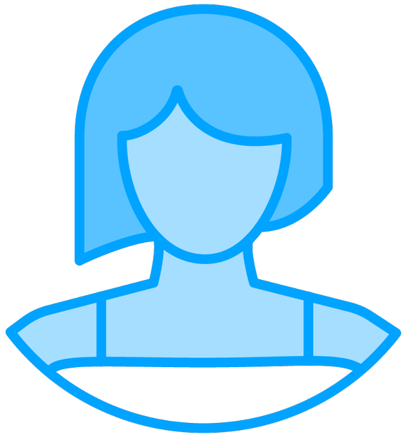
- Has a Fluent Interface

Plain HTTP Client (library or native)

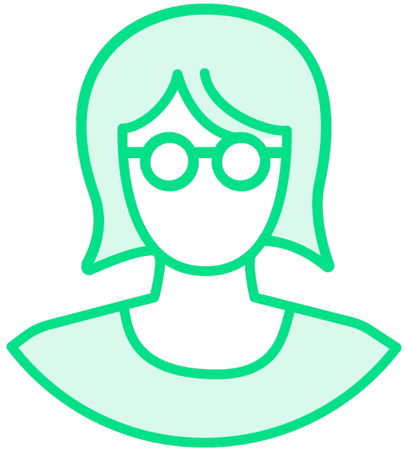
REST Assured Fluent Interface

```
RestAssured.get(“api/url”)
    .then()
    .assertThat()
        .statusCode(200)
    .and()
    .contentType(ContentType.JSON);
```

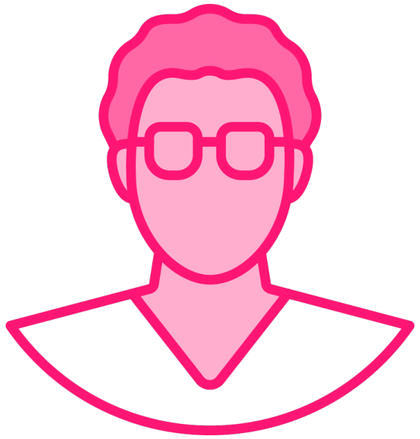
Who This Course Is For



Technical Testers



Automation Engineers



Developers

Prerequisites

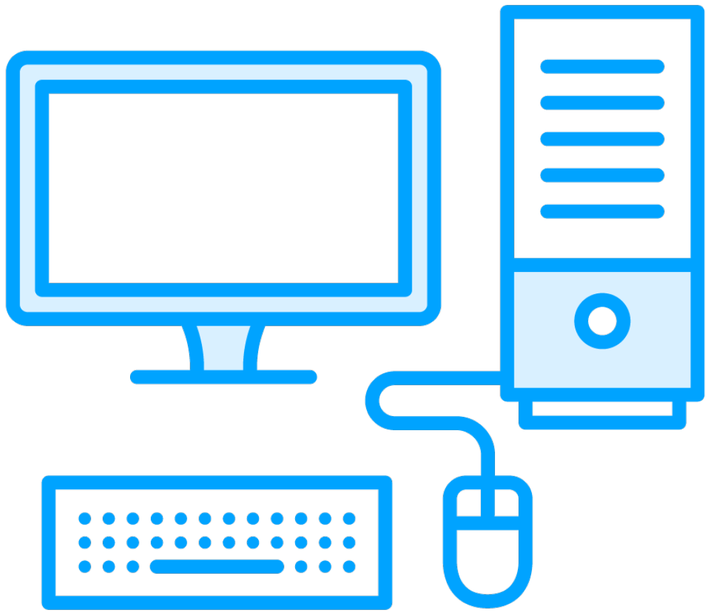
6+ months of working with Java

IDE: e.g. IntelliJ or Eclipse

Test Runner: JUnit or TestNG

Basics of the HTTP protocol

Tech Used



IntelliJ

TestNG

Maven

HTTP Refresher

Client:

- Browser
- Postman
- Java Program

Endpoint

https://api.github.com
https://api.github.com/users/{user}
https://api.github.com/users/{user}/repos

Response

Header (metadata)

e.g.
Status 200 OK
Status 404 Not Found

Body with data (payload)

Typically JSON

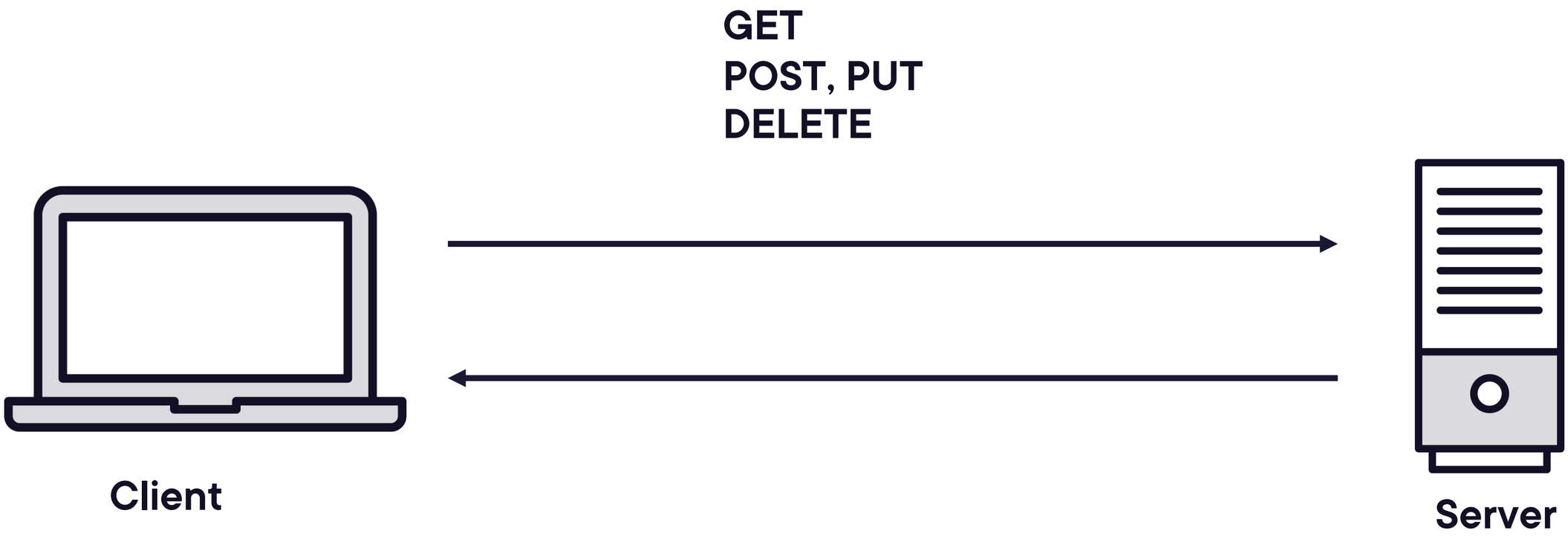
XML

```
<root>
  <login>somename</login>
  <id>12345</id>
  <followers>14</followers>
</root>
```

JSON

```
{
  login: "somename",
  id: 12345,
  followers: 14,
}
```

Other widespread HTTP methods:



Web API vs. REST API

Web API

A more general term

An API that understands HTTP

VS

REST API

A Web API...

Designed according to RESTful principles

RESTful principles

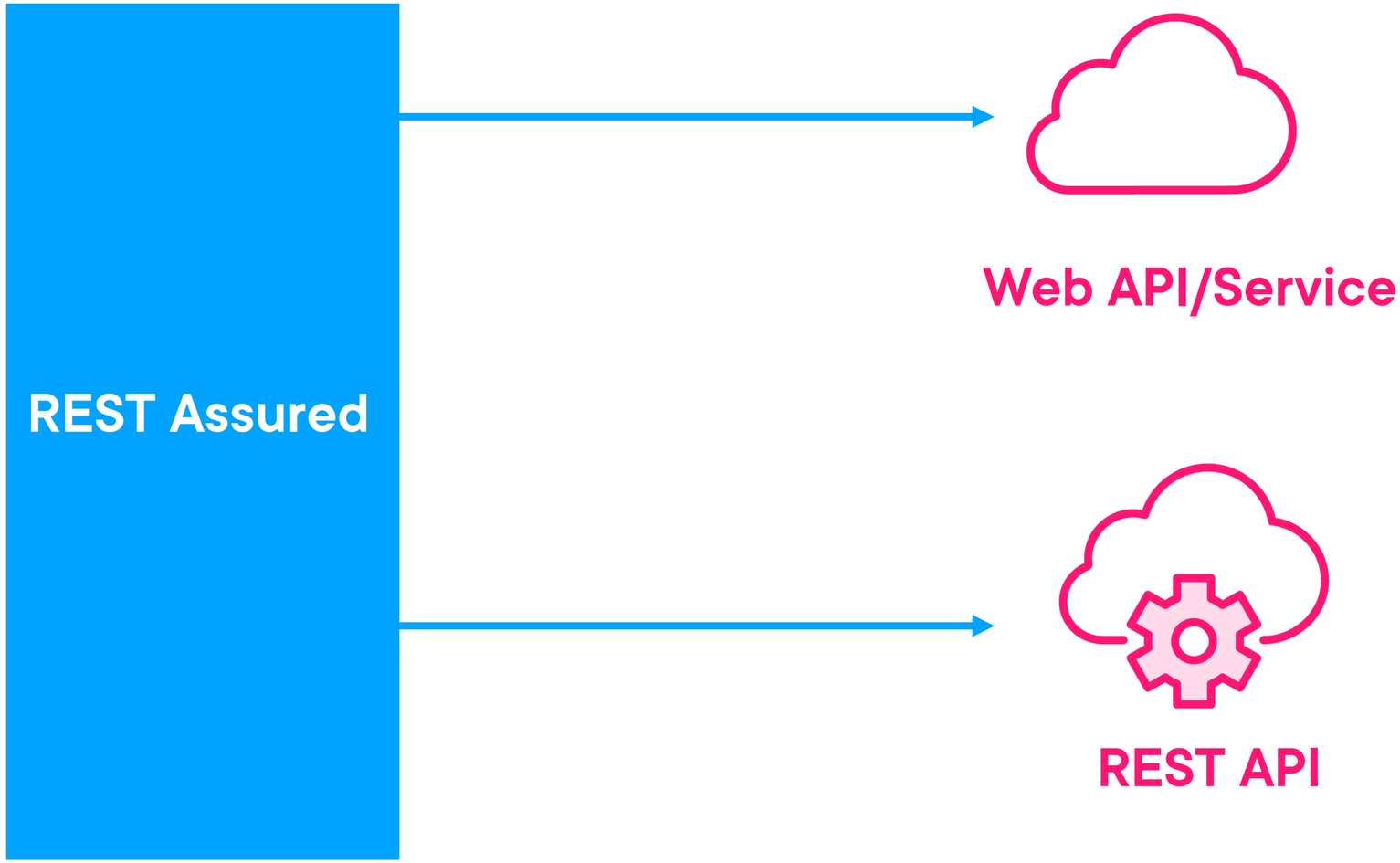
Not a protocol or a standard

Doesn’t mean we must use JSON

RESTful principles

Architectural constraints and principles:

- Uniform Interface
- Statelessness
- Layered System
- Cacheability


REST Assured
Web API/Service
REST API

REST API

A Web API...

Designed according to RESTful principles

VS

SOAP

Simple Object Access Protocol

XML-based

Imposes rules

ACID

Older

XML format

SOAP

guidelines

rules

enterprise

newer

RESTful

lightweight

usually JSON

Demo APIs

https://api.github.com/

https://jsonplaceholder.typicode.com/

https://reqres.in/

Summary

Setup

GET: Validating Response headers

GET: Validating Response body

Other HTTP methods

Config and specifications

Clean and maintainable tests