REST Assured API Testing - Complete Revision Notes for Interviews

---

✅ 1. What is REST Assured?

REST Assured is a Java library used for testing RESTful APIs. It provides a domain-specific language (DSL) for writing tests for REST services in a simple and expressive manner.

✅ 2. Basic REST Assured Syntax:

```java
import io.restassured.RestAssured;

import io.restassured.response.Response;

import static io.restassured.RestAssured.*;


Response response = given()

    .header("Content-Type", "application/json")

    .when()

    .get("https://api.example.com/users")

    .then()

    .statusCode(200)

    .extract().response();
```

✅ 3. HTTP Methods:

```java
// GET
get("/users");


// POST
given().body(jsonData).post("/users");


// PUT
given().body(updatedData).put("/users/1");


// DELETE
delete("/users/1");
```

✅ 4. Validating Response:

```java
.then().statusCode(200)
    .body("data.id", equalTo(1));
```

✅ 5. Logging Request and Response:

```java
given().log().all();
```

```
then().log().all();
```

✅ 6. Query & Path Parameters:

```java
// Path
.get("/users/{id}", 2);


// Query
.queryParam("page", 2)
.get("/users");
```

✅ 7. Authentication:

```java
// Basic
.auth().basic("username", "password");


// OAuth2
.auth().oauth2("token");
```

✅ 8. JSON/XML Parsing:

```java
response.jsonPath().get("data.name");

response.xmlPath().get("response.user.name");
```

✅ 9. Schema Validation:

```java
.then().body(JsonSchemaValidator.matchesJsonSchemaInClasspath("schema.json"));
```

✅ 10. Serialization and Deserialization:

```java
// POJO to JSON

ObjectMapper mapper = new ObjectMapper();

String json = mapper.writeValueAsString(myObject);


// JSON to POJO

MyObject obj = mapper.readValue(response.asString(), MyObject.class);
```

✅ 11. REST Principles:

- Stateless
- Client-Server

- Cacheable
- Uniform Interface
- Layered System
- Code on Demand (optional)

✅ **12. API Caching:**

- Use of cache-control, expires, etag headers.
- Helps avoid repeated API calls, improves performance.

Example:

```java
.header("If-None-Match", etagValue)
```

✅ **13. API Mocking:**

- Simulate server behavior during testing.
- Tools: WireMock, MockServer

Example:

```java
stubFor(get(urlEqualTo("/users/1"))


.willReturn(aResponse().withStatus(200).withBody("{\"name\":\"Tes
t\"}")));
```

✅ 14. Response Time Validation:

```java
.then().time(lessThan(2000L));
```

✅ 15. Parallel Execution with TestNG:

➤ By Packages:

```unset
<suite name="Parallel Test Suite" parallel="tests"
thread-count="2">

  <test name="Test1">

    <packages>

      <package name="com.api.tests.package1" />

    </packages>

  </test>

  <test name="Test2">

    <packages>

      <package name="com.api.tests.package2" />

    </packages>

  </test>

</suite>
```

➤ By Instances (Classes):

Unset

```
<suite name="Parallel Suite" parallel="instances"
thread-count="2">

  <test name="API Tests">

    <classes>

      <class name="com.api.tests.UserTest" />

      <class name="com.api.tests.ProductTest" />

    </classes>

  </test>

</suite>
```

✅ 16. Request and Response Specification:

Java

```
RequestSpecification requestSpec = new RequestSpecBuilder()

    .setBaseUri("https://api.example.com")

    .setContentType(ContentType.JSON)

    .build();


ResponseSpecification responseSpec = new ResponseSpecBuilder()

    .expectStatusCode(200)

    .expectContentType(ContentType.JSON)

    .build();
```

```
// Usage:
given().spec(requestSpec)
.when().get("/users")
.then().spec(responseSpec);
```

✅ **17. Handling Dynamic Parameters:**

```
Java
Map<String, Object> params = new HashMap<>();
params.put("userId", 1);
params.put("type", "active");

given().params(params).get("/users");
```

✅ **18. Rate Limiting / Throttling:**

- Usually returns status code 429 (Too Many Requests)
- Use retry logic or wait time

```
Java
if(response.statusCode() == 429) {
    Thread.sleep(5000);
```

```
    // Retry the request
}
```

✅ 19. BDD Style:

```java
given()
  .when().get("/users")
  .then().assertThat().statusCode(200);
```

✅ 20. File Upload and Download:

➤ File Upload:

```java
File file = new File("src/test/resources/sample.pdf");


given()
    .multiPart("file", file)
    .when()
    .post("/upload")
    .then().statusCode(200);
```

➤ File Download:

```java
Response response = given()
    .when()
    .get("/download/sample.pdf");

byte[] fileData = response.asByteArray();
Files.write(Paths.get("downloaded_sample.pdf"), fileData);
```

✅ **21. Form Parameters:**

```java
given()
    .formParam("username", "admin")
    .formParam("password", "pass123")
    .when()
    .post("/login")
    .then().statusCode(200);
```

✅ **22. Upstream vs Downstream APIs:**

- **Upstream API**: External system your service depends on. You call it.
- **Downstream API**: API that depends on your service. It calls your API.

Interview Context:

- How do you mock an upstream API?
- How do you ensure backward compatibility for downstream consumers?

✅ 23. Common Interview Questions:

1. What is REST Assured? How is it better than other tools?
2. How do you validate response codes and body?
3. Explain serialization and deserialization in REST Assured.
4. How do you perform schema validation?
5. How do you test APIs that require tokens/authentication?
6. What are REST constraints?
7. How to mock an API in real-time testing?
8. What is the use of caching in APIs?
9. How do you handle dynamic parameters in REST Assured?
10. How to achieve parallel execution in TestNG?
11. Difference between request specification and response specification?
12. How do you test rate limiting or throttling scenarios?
13. Explain BDD style in REST Assured.
14. How to handle file upload or multipart/form-data in REST Assured?
15. How to validate response headers?
16. How to pass bearer token dynamically in tests?
17. How to reuse token/authentication in multiple tests?
18. What are upstream and downstream APIs?

19. How to download and store a file from an API?

20. How to handle form-based login authentication?

This doc gives you **100% coverage** for REST Assured-based interview preparation. Use it as your quick revision go-to guide before interviews. ✅