

Serializing Data using Java Class

Used jackson data binding to help in serializing the object as a JSON body.

HTTP Response Body:

Consists of:

- 1) HTTP Status Code
- 2) Headers
- 3) Response Body

Headers contains info about:

- 1) Cookies
- 2) Connection type
- 3) Proxies
- 4) Pagination etc.

Assertions in Rest Assured

Uses hamcrest matchers. Helps to compare actual results with expected results.

Examples:

- 1) notNull
- 2) greaterThan
- 3) lessThan
- 4) greaterThanOrEqualTo
- 5) equalTo (matches the memory locations of the objects instead of actual values)
- 6) samePropertyValueAs() etc.

Headers

Content type is critical to validate for APIs because if it's not correct then consumers attempting to parse the response might break.

Deserializing the Response Body:

When modeling the API response using a java class, it's called de-serializing.

Q1) Why would you want to deserialize an API response?

Ans: to model a body and represent it as a Java object

Q2) You have an API response and need to verify that xml data is being returned. What is the correct way to do this?

Ans: `header("Content-Type", equalTo("application/xml; charset=UTF-8"))`

Q3) When examining an API response, which component is the most critical to validate?

Ans: the body

Q4) What is the correct statement to verify a status code of 200?

Ans: `given()`.

```
queryParams("id",2).  
when().get(endpoint).  
then().assertThat().statusCode(200);
```

Q5) How can you print the body of an API response to the console?

Ans:

```
public void printProduct() {  
String endpoint = "http://localhost:8888/api_testing/product/read_one.php";  
given().queryParams("id", 2).  
when().get(endpoint).  
then().log().body(); }
```

Q6) You want to verify that category 2 has the name "active wear". Which assertThat() statement is correct?

Ans: assertThat().
statusCode(200).

```
body("category_id", equalTo("2")).  
body("category_name", equalTo("active wear"))
```

Q7) How can you assure you have records in your API response, and each record has an id that exists?

Ans: assertThat().
statusCode(200).
body("records.size()", greaterThan(0)).
body("records.id", everyItem(notNullValue()))

Q8) Which method directly precedes the assertThat() method?

Ans: then()

#Why Static Keywords in Rest Assured?

- Helps in better code readability
- If we do not use static keywords, then we need to use ClassNames. Which reduces the code readability.
- Which increases the number of lines of code.
- If we do not use, given(), when(), then(). Then we do not need to use static imports.

#How does Method Chaining work?

Given(), When(), Then() -> return an object of the same class and implements RequestSpecification Interface. Since all the methods return the same object, it is possible to chain them.

Hamcrest Matcher Assertion Deep Dive:

-> Hamcrest is a well known assertion library used for unit testing along with JUnit.

- > Hamcrest can be used along with Rest Assured for assertions.
- > Uses matcher classes for making assertions

Adv:

- > Human readable and in plain english
- > Code is neat and intuitive

- > Provides thin methods like "is" and "not", also called as decorators, for more readability

Hamcrest Vs TestNG

- > Readability
- > Descriptive error messages
- > Type Safety

Collection matchers (List, Array, Map, etc.)

=====

- hasItem()** -> check single element in a collection
- not(hasItem())** -> check single element is NOT in a collection
- hasItems()** -> Check all elements are in a collection
- contains()** -> Check all elements are in a collection and in a strict order
- containsInAnyOrder()** -> Check all elements are in a collection and in any order
- empty()** -> Check if collection is empty
- not(emptyArray())** -> Check if the Array is not empty
- hasSize()** -> Check size of a collection
- everyItem(startsWith())** -> Check if every item in a collection starts with specified string
- hasKey()** -> Map -> Check if Map has the specified key [value is not checked]
- hasValue()** -> Map -> Check if Map has at least one key matching specified value
- hasEntry()** -> Maps -> Check if Map has the specified key value pair
- equalTo(Collections.EMPTY_MAP)** -> Maps [Check if empty]
- allOf()** -> Matches if all matches matches
- anyOf()** -> Matches if any of the matches matches

Numbers:

- greaterThanOrEqualTo()**
- lessThan()**
- lessThanOrEqualTo()**

String:

- containsString()**
- emptyString()**

Logging Mechanism in Rest Assured

- 1) **log().all():** Logs the entire data

Can be used with Given() and Then(). Meaning Request Data and Response Data.

- 2) **log().headers():** Logs the headers present in Request and Response Data.

Can be used with given() & then()

3) **log().body()**: Helps in validating or reviewing the response body.

Should be used with `then()`.

4) **log().ifError()**: works only when there is an error in response while validating the response.

Should be used with `then()`.

5) **log().ifValidationFails()**: Works when assertions fail during response validations.