Request Specification in Java Rest Assured :-

Request Specification in Rest Assured is a powerful interface that allows you to define a set of conditions or configurations for HTTP requests that you plan to send. It helps you to avoid repetitive code by allowing you to predefine request settings like base URIs, headers, authentication, parameters, cookies, and more. Once you define these settings in a RequestSpecification object, you can reuse them across multiple requests.

Request Specification in Rest Assured allows you to define and reuse common settings across multiple API requests. This is particularly useful when you have a set of requests that share the same configurations, such as headers, base URI, base path, authentication, and more. By using Request Specification, you avoid redundancy in your code, making it cleaner, more maintainable, and less error-prone.

Advantages:

1. **Reusability:** You can define a request specification once and reuse it across multiple tests, which reduces code duplication.
2. **Maintainability:** If there's a change in the common request configuration, you only need to update the request specification, not each individual test.
3. **Readability:** Using a request specification makes your test cases cleaner and easier to read, as the common configurations are abstracted away.
4. **Modularity:** You can break down your request configurations into reusable components, making your tests more modular and easier to manage.

A **Request Specification** is an interface in Rest Assured that allows you to specify how the request should be constructed. It includes settings like base URI, base path, headers, cookies, query parameters, form parameters, request body, authentication, and more. Once defined, this specification can be used to create a request without having to redefine these settings every time.

RequestSpecBuilder :

The RequestSpecBuilder class in Rest Assured provides a builder pattern to create RequestSpecification instances. It allows you to define a request specification step by step and then build it. This approach is particularly useful when dealing with complex request configurations.

Example of Using RequestSpecBuilder

```
RequestSpecification requestSpec = new RequestSpecBuilder()
    .setBaseUri("https://api.example.com")
```

```
.setBasePath("/v1")
.addHeader("Authorization", "Bearer token")
.setContentType(ContentType.JSON)
.log(LogDetail.ALL)
.build();
```

Key Methods in Request Specification

1. **baseUri(String uri)**
    - **Description:** Sets the base URI for the request.
    - **Return Type:** RequestSpecification
2. **basePath(String path)**
    - **Description:** Sets the base path for the request.
    - **Return Type:** RequestSpecification
3. **header(String name, Object value)**
    - **Description:** Adds a single header to the request.
    - **Return Type:** RequestSpecification
4. **headers(Map<String, ?> headersMap)**
    - **Description:** Adds multiple headers to the request.
    - **Return Type:** RequestSpecification
5. **param(String name, Object value)**
    - **Description:** Adds a query parameter to the request.
    - **Return Type:** RequestSpecification
6. **params(Map<String, ?> parametersMap)**
    - **Description:** Adds multiple query parameters to the request.
    - **Return Type:** RequestSpecification
7. **queryParam(String name, Object value)**
    - **Description:** Adds a query parameter to the request URL.

- **Return Type:** RequestSpecification

8. **queryParams(Map<String, ?> parametersMap)**
    - **Description:** Adds multiple query parameters to the request URL.
    - **Return Type:** RequestSpecification
9. **formParam(String name, Object value)**
    - **Description:** Adds a form parameter to the request body.
    - **Return Type:** RequestSpecification
10. **formParams(Map<String, ?> parametersMap)**
    - **Description:** Adds multiple form parameters to the request body.
    - **Return Type:** RequestSpecification
11. **body(Object body)**
    - **Description:** Sets the request body.
    - **Return Type:** RequestSpecification
12. **auth()**
    - **Description:** Returns an object to define authentication settings for the request.
    - **Return Type:** AuthenticationSpecification
13. **contentType(ContentType contentType)**
    - **Description:** Specifies the content type of the request.
    - **Return Type:** RequestSpecification
14. **accept(ContentType contentType)**
    - **Description:** Specifies the expected content type of the response.
    - **Return Type:** RequestSpecification
15. **cookie(String name, Object value)**
    - **Description:** Adds a cookie to the request.
    - **Return Type:** RequestSpecification
16. **cookies(Map<String, ?> cookiesMap)**
    - **Description:** Adds multiple cookies to the request.

- **Return Type:** RequestSpecification

17. **log()**
    - **Description:** Returns an object to specify logging settings for the request.
    - **Return Type:** LogSpecification
18. **relaxedHTTPSValidation()**
    - **Description:** Bypasses SSL certificate validation.
    - **Return Type:** RequestSpecification
19. **when()**
    - **Description:** Marks the start of the request and returns a RequestSender.
    - **Return Type:** RequestSender
20. **multiPart(String controlName, String fileName, byte[] bytes)**
    - **Description:** Adds a multi-part form data.
    - **Return Type:** RequestSpecification
21. **sessionId(String sessionId)**
    - **Description:** Adds a session ID to the request.
    - **Return Type:** RequestSpecification
22. **filter(Filter filter)**

    Description: Adds a filter to the request.
    - **Return Type:** RequestSpecification

23. **then()**
    - **Description:** Marks the starting point of the response assertions.
    - **Return Type:** ResponseSpecification

Example 1: Setting Base URI and Sending a GET Request

```
RequestSpecification requestSpec = given()
    .baseUri("https://jsonplaceholder.typicode.com");
```

```
Response response = requestSpec.when().get("/posts/1");

response.prettyPrint();
```

Explanation:

1. A RequestSpecification object is created with the base URI set to "https://jsonplaceholder.typicode.com".
2. A GET request is made to the /posts/1 endpoint.
3. The response is printed.

Example 2: Adding Headers and Sending a POST Request

```
RequestSpecification requestSpec = given()
    .baseUri("https://jsonplaceholder.typicode.com")
    .header("Content-Type", "application/json");


Response response = requestSpec.when().post("/posts");

response.prettyPrint();
```

Explanation:

1. A RequestSpecification object is created with the base URI and a Content-Type header.
2. A POST request is made to the /posts endpoint.
3. The response is printed.

Example 3: Adding Query Parameters to a GET Request

```
RequestSpecification requestSpec = given()
    .baseUri("https://jsonplaceholder.typicode.com")
    .queryParam("userId", 1);


Response response = requestSpec.when().get("/posts");

response.prettyPrint();
```

Explanation:

1. A RequestSpecification object is created with the base URI and a query parameter userId.

2. A GET request is made to the /posts endpoint with the query parameter.
3. The response is printed.

Example 4: Sending a Request with Form Parameters

```
RequestSpecification requestSpec = given()
   .baseUri("https://jsonplaceholder.typicode.com")
   .formParam("title", "foo")
   .formParam("body", "bar");

Response response = requestSpec.when().post("/posts");
response.prettyPrint();
```

Explanation:

1. A RequestSpecification object is created with the base URI and form parameters.
2. A POST request is made to the /posts endpoint with the form parameters.
3. The response is printed.

Example 5: Setting a Session ID and Sending a GET Request

```
RequestSpecification requestSpec = given()
   .baseUri("https://jsonplaceholder.typicode.com")
   .sessionId("session123");

Response response = requestSpec.when().get("/posts/1");
response.prettyPrint();
```

Explanation:

1. A RequestSpecification object is created with the base URI and a session ID.
2. A GET request is made to the /posts/1 endpoint.
3. The response is printed.

Example 6: Configuring Authentication with Basic Auth

```
RequestSpecification requestSpec = given()
```

```
    .baseUri("https://jsonplaceholder.typicode.com")
    .auth().basic("username", "password");

Response response = requestSpec.when().get("/posts/1");
response.prettyPrint();
```

Explanation:

1. A RequestSpecification object is created with basic authentication.
2. A GET request is made to the /posts/1 endpoint.
3. The response is printed.

Example 7: Adding Multiple Headers and Query Parameters

```
RequestSpecification requestSpec = given()
    .baseUri("https://jsonplaceholder.typicode.com")
    .headers("Content-Type", "application/json", "Accept", "application/json")
    .queryParams("userId", 1, "id", 1);

Response response = requestSpec.when().get("/posts");
response.prettyPrint();
```

Explanation:

1. Multiple headers and query parameters are added to the request.
2. A GET request is made to the /posts endpoint.
3. The response is printed.

Example 8: Sending JSON Body with POST Request

```
RequestSpecification requestSpec = given()
    .baseUri("https://jsonplaceholder.typicode.com")
    .header("Content-Type", "application/json")
    .body("{\"title\":\"foo\",\"body\":\"bar\",\"userId\":1}");
```

```
Response response = requestSpec.when().post("/posts");
response.prettyPrint();
```

Explanation:

1. The request body is set with a JSON payload.
2. A POST request is made to the /posts endpoint.
3. The response is printed.

Example 9: Using Cookies in a GET Request

```
RequestSpecification requestSpec = given()
    .baseUri("https://jsonplaceholder.typicode.com")
    .cookie("session_id", "abc123");

Response response = requestSpec.when().get("/posts/1");
response.prettyPrint();
```

Explanation:

1. A cookie is added to the request.
2. A GET request is made to the /posts/1 endpoint.
3. The response is printed.

Example 10: Using Multi-Part Form Data

```
RequestSpecification requestSpec = given()
    .baseUri("https://jsonplaceholder.typicode.com")
    .multiPart("file", new File("path/to/file"));

Response response = requestSpec.when().post("/upload");
response.prettyPrint();
```

Explanation:

1. A file is added as multi-part form data.
2. A POST request is made to the /upload endpoint.

3. The response is printed.

Example 11: Setting Base URI and Common Headers

```
import io.restassured.RestAssured;
import io.restassured.specification.RequestSpecification;
import static io.restassured.RestAssured.given;

public class Example1 {
   public static void main(String[] args) {
      // Step 1: Create Request Specification
      RequestSpecification requestSpec = RestAssured.given()
            .baseUri("https://api.example.com")
            .header("Accept", "application/json")
            .header("User-Agent", "RestAssured");

      // Step 2: Make the GET Request using the specification
      given().spec(requestSpec)
            .when().get("/users")
            .then().statusCode(200);
   }
}
```

Explanation:

- baseUri() sets the base URI for the requests.
- header() adds common headers that are used in all requests.
- The request is made using the spec(requestSpec) method.

Example 12: Using Query Parameters and Logging

```
import io.restassured.RestAssured;
import io.restassured.specification.RequestSpecification;
```

```
import static io.restassured.RestAssured.given;

public class Example2 {
    public static void main(String[] args) {
        // Step 1: Create Request Specification
        RequestSpecification requestSpec = RestAssured.given()
                .baseUri("https://api.example.com")
                .queryParam("status", "active")
                .log().all();

        // Step 2: Make the GET Request using the specification
        given().spec(requestSpec)
                .when().get("/users")
                .then().statusCode(200);
    }
}
```

Explanation:

- queryParam() adds a query parameter to the request.
- log().all() logs all the details of the request.

Example 13: Setting Content Type and Body

```
import io.restassured.RestAssured;
import io.restassured.specification.RequestSpecification;
import static io.restassured.RestAssured.given;
import io.restassured.http.ContentType;

public class Example3 {
    public static void main(String[] args) {
```

```
        // Step 1: Create Request Specification
        RequestSpecification requestSpec = RestAssured.given()
                .baseUri("https://api.example.com")
                .contentType(ContentType.JSON)
                .body("{ \"name\": \"Mahii\", \"age\": 25 }")
                .log().all();

        // Step 2: Make the POST Request using the specification
        given().spec(requestSpec)
                .when().post("/users")
                .then().statusCode(201);
    }
}
```

Explanation:

- contentType() sets the content type of the request to JSON.
- body() sets the body of the request.

Example 14: Using Form Parameters for a POST Request

```
import io.restassured.RestAssured;
import io.restassured.specification.RequestSpecification;
import static io.restassured.RestAssured.given;

public class Example4 {
    public static void main(String[] args) {
        // Step 1: Create Request Specification
        RequestSpecification requestSpec = RestAssured.given()
                .baseUri("https://api.example.com")
                .formParam("username", "Mike")
```

```
            .formParam("password", "password123")
            .log().all();

        // Step 2: Make the POST Request using the specification
        given().spec(requestSpec)
            .when().post("/login")
            .then().statusCode(200);
    }
}
```

Explanation:

- formParam() is used to add form parameters for the POST request.

Example 15: Authentication and Sending Custom Headers

```
import io.restassured.RestAssured;
import io.restassured.specification.RequestSpecification;
import static io.restassured.RestAssured.given;

public class Example5 {
    public static void main(String[] args) {
        // Step 1: Create Request Specification
        RequestSpecification requestSpec = RestAssured.given()
            .baseUri("https://api.example.com")
            .auth().basic("username", "password")
            .header("X-Custom-Header", "CustomValue")
            .log().all();

        // Step 2: Make the GET Request using the specification
        given().spec(requestSpec)
```

```
            .when().get("/secure-data")
            .then().statusCode(200);
    }
}
```

Explanation:

- auth().basic() is used for basic authentication.
- header() is used to add a custom header to the request.

Example 16: Basic Request with Reusable Configuration

```
import io.restassured.RestAssured;
import io.restassured.specification.RequestSpecification;
import static io.restassured.RestAssured.given;

public class ApiTest {
    public static void main(String[] args) {
        RequestSpecification requestSpec = RestAssured.given()
            .baseUri("https://api.example.com")
            .header("Authorization", "Bearer abc123")
            .contentType("application/json");

        // Use the request specification to send a GET request
        given()
            .spec(requestSpec)
        .when()
            .get("/users/1")
        .then()
```

```
            .statusCode(200)
            .log().all();
    }
}
```

Explanation:

- The RequestSpecification is created with a base URI, a header for authorization, and a content type.
- This specification is reused in the given() method for sending a GET request to retrieve user data.

Example 17: Adding Query Parameters to Request

```
import io.restassured.specification.RequestSpecification;
import static io.restassured.RestAssured.given;

public class ApiTest {
    public static void main(String[] args) {
        RequestSpecification requestSpec = given()
            .baseUri("https://api.example.com")
            .header("Authorization", "Bearer abc123")
            .contentType("application/json");

        // Adding query parameters dynamically
        given()
            .spec(requestSpec)
            .queryParam("status", "active")
            .queryParam("role", "admin")
        .when()
            .get("/users")
        .then()
```

```
            .statusCode(200)
            .log().all();
    }
}
```

Explanation:

- In this example, query parameters are added to filter the list of users by status and role.
- The request specification is reused, and the query parameters are added in the given() method.

Example 18: Form Parameter Submission

```
import io.restassured.specification.RequestSpecification;
import static io.restassured.RestAssured.given;

public class ApiTest {
    public static void main(String[] args) {
        RequestSpecification requestSpec = given()
            .baseUri("https://api.example.com")
            .header("Authorization", "Bearer abc123")
            .contentType("application/x-www-form-urlencoded");

        // Submitting form parameters
        given()
            .spec(requestSpec)
            .formParam("username", "Sweta")
            .formParam("password", "password123")
        .when()
            .post("/login")
        .then()
```

```
            .statusCode(200)
            .log().all();
    }
}
```

Explanation:

- This example demonstrates submitting form parameters using the formParam() method.
- The content type is set to application/x-www-form-urlencoded to match the form submission format.

Example 19: Using Cookies and Logging

```
import io.restassured.specification.RequestSpecification;
import static io.restassured.RestAssured.given;

public class ApiTest {
    public static void main(String[] args) {
        RequestSpecification requestSpec = given()
            .baseUri("https://api.example.com")
            .header("Authorization", "Bearer abc123")
            .cookie("sessionId", "xyz987")
            .log().all();

        // Sending a request with cookies and logging enabled
        given()
            .spec(requestSpec)
        .when()
            .get("/account")
        .then()
            .statusCode(200)
```

```
        .log().all();
    }
}
```

Explanation:

- This example shows how to add cookies to the request and enable logging for both the request and response.
- The session ID is sent as a cookie, which might be required for authentication.

Example 20: Using Authentication and Base Path

```
import io.restassured.RestAssured;
import io.restassured.specification.RequestSpecification;
import static io.restassured.RestAssured.given;

public class ApiTest {
    public static void main(String[] args) {
        RequestSpecification requestSpec = RestAssured.given()
            .baseUri("https://api.example.com")
            .basePath("/v1")
            .auth().preemptive().basic("Sweta", "password123")
            .contentType("application/json");

        // Sending a request with basic authentication and base path
        given()
            .spec(requestSpec)
        .when()
            .get("/users/1")
        .then()
            .statusCode(200)
```

```
        .log().all();
    }
}
```

Explanation:

- This example demonstrates using basic authentication and setting a base path for the API.
- The auth().preemptive().basic() method is used for basic authentication with the username "Sweta" and password "password123".

The base path is set to /v1, so all requests using this specification will automatically include this path.

Example 21: Complex Authentication and Dynamic Headers

Scenario: Authenticate using OAuth2 and add dynamic headers based on the response of the authentication API.

```
// Step 1: Obtain OAuth2 token
Response authResponse = given()
    .auth()
    .preemptive()
    .basic("clientId", "clientSecret")
    .formParam("grant_type", "client_credentials")
    .post("https://auth.example.com/oauth/token");

String token = authResponse.jsonPath().getString("access_token");

// Step 2: Build Request Specification with dynamic headers
RequestSpecification requestSpec = new RequestSpecBuilder()
    .setBaseUri("https://api.example.com")
    .addHeader("Authorization", "Bearer " + token)
    .addHeader("Dynamic-Header", "HeaderValue")
```

```
    .setContentType(ContentType.JSON)
    .log(LogDetail.ALL)
    .build();

// Step 3: Use the Request Specification
given()
    .spec(requestSpec)
    .when()
    .get("/data")
    .then()
    .statusCode(200);
```

Explanation:

- Authenticate using OAuth2 and retrieve a token.
- Build a RequestSpecification with dynamic headers based on the retrieved token.
- Make a GET request using the built specification.

Example 22: Complex Query Parameter Handling

Scenario: Make a request with a large number of query parameters, some of which are dynamically generated.

```
// Step 1: Build dynamic query parameters
Map<String, String> queryParams = new HashMap<>();
queryParams.put("param1", "value1");
queryParams.put("param2", "value2");
// Add more parameters as needed

// Step 2: Build Request Specification
RequestSpecification requestSpec = new RequestSpecBuilder()
    .setBaseUri("https://api.example.com")
    .addQueryParams(queryParams)
```

```
    .addQueryParam("dynamicParam", System.currentTimeMillis()) // Dynamic value
    .setContentType(ContentType.JSON)
    .log(LogDetail.ALL)
    .build();

// Step 3: Use the Request Specification
given()
    .spec(requestSpec)
    .when()
    .get("/search")
    .then()
    .statusCode(200);
```

Explanation:

- Create a map of query parameters, including dynamic values.
- Build a RequestSpecification with the query parameters.
- Use the specification to make a GET request.

Example 23: Advanced Multi-Part File Upload

Scenario: Upload a file along with additional form parameters and headers.

```
// Step 1: Build Request Specification for multi-part upload
RequestSpecification requestSpec = new RequestSpecBuilder()
    .setBaseUri("https://api.example.com")
    .addHeader("Authorization", "Bearer token")
    .setContentType(ContentType.MULTIPART)
    .log(LogDetail.ALL)
    .build();

// Step 2: Use the Request Specification to upload the file
given()
```

```
    .spec(requestSpec)
    .multiPart("file", new File("path/to/file.txt"))
    .formParam("param1", "value1")
    .formParam("param2", "value2")
    .when()
    .post("/upload")
    .then()
    .statusCode(201);
```

Explanation:

- Create a RequestSpecification with multi-part content type and headers.
- Use the specification to upload a file with additional form parameters.

Example 24: Handling Complex Nested JSON Request Body

Scenario: Send a POST request with a complex nested JSON body.

```
// Step 1: Create a complex JSON object
Map<String, Object> nestedObject = new HashMap<>();
nestedObject.put("field1", "value1");
nestedObject.put("field2", Arrays.asList("item1", "item2"));

Map<String, Object> requestBody = new HashMap<>();
requestBody.put("mainField", nestedObject);
requestBody.put("otherField", "value");

// Step 2: Build Request Specification
RequestSpecification requestSpec = new RequestSpecBuilder()
    .setBaseUri("https://api.example.com")
    .setContentType(ContentType.JSON)
    .log(LogDetail.ALL)
```

```
    .build();


// Step 3: Use the Request Specification to send the POST request
given()
    .spec(requestSpec)
    .body(requestBody)
    .when()
    .post("/create")
    .then()
    .statusCode(201);
```

Explanation:

- Create a complex nested JSON object as the request body.
- Build a RequestSpecification with JSON content type.
- Use the specification to send a POST request.

Example 25: Advanced Request with Multiple Cookies and Authentication

Scenario: Make a request with multiple cookies and digest authentication.

```
// Step 1: Build Request Specification with cookies and authentication
RequestSpecification requestSpec = new RequestSpecBuilder()
    .setBaseUri("https://api.example.com")
    .addCookie("cookie1", "value1")
    .addCookie("cookie2", "value2")
    .setAuth(digest("username", "password"))
    .setContentType(ContentType.JSON)
    .log(LogDetail.ALL)
    .build();


// Step 2: Use the Request Specification to send a GET request
given()
```

```
.spec(requestSpec)
.when()
.get("/secure-data")
.then()
.statusCode(200);
```

Explanation:

- Build a RequestSpecification with multiple cookies and digest authentication.
- Use the specification to send a GET request to a secure endpoint.

These examples cover various aspects of using RequestSpecification in Rest Assured, from simple headers and query parameters to more complex scenarios like form submission, cookies, and authentication.