

Java object, it can be POJO, Map, List, it is converted to JSON or XML object and this process is called - **serialization**,

and it is done by different libraries (Jakson, Gson for JSON, JAXB-for xml object) these libraries are called as - serializers. Once the Java object is serialized into JSON or XML object then it is sent as a request payload as part of an HTTP request that goes to the server. The server then process the request and sends back HTTP response.

The HTTP response might have a response body, which can be JSON or XML object. Then this JSON or XML object converted back to Java object and this process is called - de **serialization**.

The process of serialization and de serialization is called - **object mapping**.

Why we are doing it while working with Rest Assured ?

Because dealing directly with JSON or XML object might be challenging. The JSON and XML object kind of technical representation of the object, while the Java object (especially POJO) representation will be simple. And since we are using the Java programming language, representing an object as a Java object brings a lot of benefits. We can easily create objects, fetch objects, maintain them, and re-use them and in case if it is POJO it is readable as well.

The important note is that serialization and de serialization is done internally by Rest Assured and it is uses one of these libraries under the hood.

**!** Для сериализации нужен JSON serializer, именно для этого мы добовляем библиотеку

<dependency>

<groupId>com.fasterxml.jackson.core</groupId>

<artifactId>jackson-databind</artifactId>

<version>2.14.1</version>

</dependency>

если этого не сделать то нельзя работать с POJO, HashMap, будет ошибка, так как Rest Assured не знает как сделать сериализацию используя эти типы данных

Ещё пример по сериализации

Например надо в запросе (в соответствии с API) отправить JSON в этом случае мы можем JSON записать в виде:

1-String

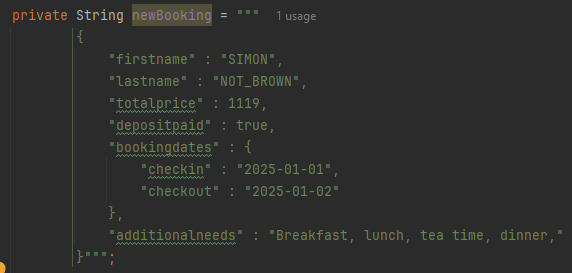
2-создать POJO класс

3-или с помощью класса JSONObject (для этого нужна dependecy в maven repo надо набрать “JSON In Java”), 4-или как HashMap

5- или как File.

Какой бы вариант мы бы не выбрали, мы должны передать этот обьект который мы собераемся отправить как аргумент метода body()

1)- Использовать стринг можно для каких то быстрых реквестов и в тренировочном процессе, чтобы понять весь процесс целиком. Но для тестирования наверно это не катит.

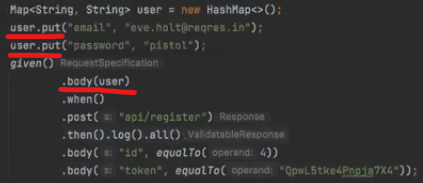


2)- POJO это самый оптимальный вариант, так как мы можем использовать один POJO класс и для реквеста и для респонса. А так же получив респонс в виде POJO легко достать любые поля для проверак (assert-ов).

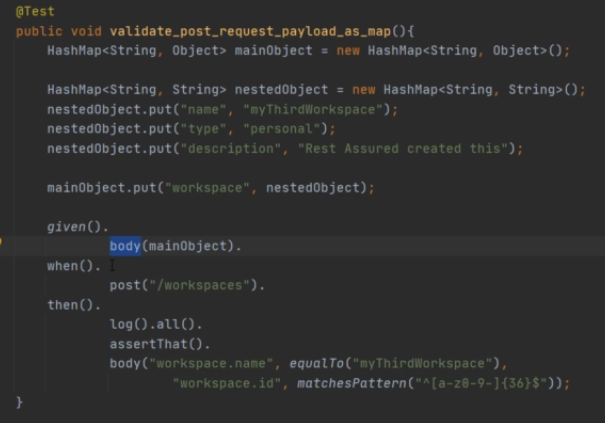
3)- класс JSONObject слишком мне кажется сложным, не уверен, что это практично.



4)Использование для сериализации HashMap. Убедись что есть JSONserializer (jackson-databind)

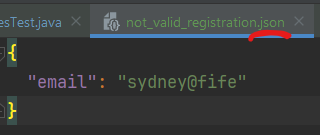


или HashMap с вложенным JSON-oм

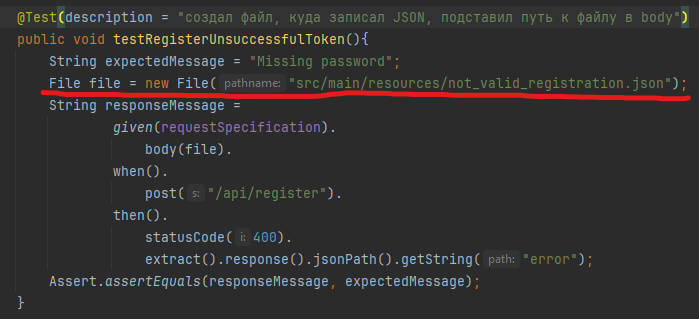


5) - Использование для сериализации File

a)Создать файл с расширением .json, запомнить путь к нему.



б) В тесте, где собераемся использовать файл надо создать обьект от класса File, в аргументы которого передать путь где лежит JSON файл



в) Передать созданный обьект класса File в body(file) или в RequestSpecification, оба варианта будут правильными.

