JsonPath is a class in RestAssured that simplifies navigating and querying JSON data.

Нам надо как то анализировать результат, а результат прилетает в виде JSON-a и у него есть нетолько разные поля но и типы значений у этих полей разные: String, int, Double…

Класс jsonPath() - работает с response (хм. похоже не только).

После того как создал обьект от JsonPath класса, можно вызвать метод jsonPath() - который позволяет в зависимости от типа значения в JSON-е применить соответствующий метод из своего арсенала, чтобы получить нужный тип значений.

**Синтаксис в jsonPath()** (от Rest Assured, есть и другие/похожие аналоги)

1) - Стоит помнить про синтаксис ‘путь’ в JSON-е к искомому полю.

**String checkin = response.jsonPath().get("bookingdates");**

2) - если обьект (поле обьекта) вложенные, надо использовать точечную нотацию.

**String checkin = response.jsonPath().get("bookingdates.checkin");**

или

**List<String> lstFirstName = jsonPath.getList("data.first\_name");**

3) - Также если JSON содержит массив надо использовать соответствующий синтаксис

**jsonPath.getString("store.book[0].category");**

4) - Last element of array can be accessed using index -1. There are 4 books in book array so, to navigate last book either we can use book[3] or book[-1]

String strLastBookCategoryUsingMinusOne = jsonPath.getString("**store.book[-1].author**"); //Output : J. R. R. Tolkien

или по другому, но с тем же эффектом.

String strLastBookCategoryUsing3 = jsonPath.getString("**store.book[3].author**"); //Output : J. R. R. Tolkien

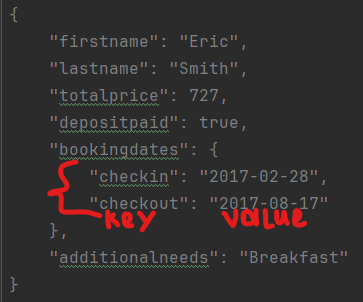
5) - getList() method is used to retrieve arrays of values, objects. To extract a list of categories for all books (from a JSON array), you can use the getList() method. This method extracts and returns a List<T>, where T depends on the data type of the elements in the array. Since value of category is a string, it will return a List<String>.

**List<String> lst = jsonPath.getList("store.book.category");**

6) - The getMap() method in RestAssured is used to extract a JSON object as a Map<String, Object> from the response. The key in the Map represents the field names (or property names) from the JSON, while the value represents the corresponding data (which could be a String, Integer, Double, nested Map, or List, depending on the structure).

Say, we like to fetch first book from array of the books. Here is the code.

**Map<String, List<String>> map = jsonPath.getMap("bookingdates");**

****

**ВНИМАНИЕ!** ниже пример с респонсом и респонс содержит JSON обьект полученный от сервера. Видимо по этому можно обращатся к методу **jsonPath()**, при этом не создавая обьект класса JsonPath.

Например: № 1

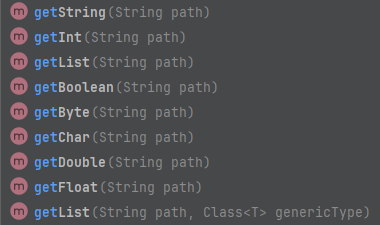
**String checkin = response.jsonPath().getString("firstname");**

**int totalPrice = response.jsonPath().getInt("totalprice");**

**boolean depositpaid = response.jsonPath().getBoolean("depositpaid");**

№ 2

Но если респонс записать в стринг, то её(стринг с респонсом в виде JSON) можно использовать для создания объекта от Класса jsonPath(). А для чего это всё? Если использовать стринг как обьект для проверок (т.е. для Assert) то придётся писать (пользоватся) сложные методы по извлечению нужных полей и их значений из этой стринг. Но если это будет обьект от Класса jsonPath() то можно вызвать метод jsonPath() (и класс и метод в этом классе одинаково называются) и легко достать любое поле используя соответствующий метод.



String stringResponse = RestAssured.get("https://restful-booker.herokuapp.com/booking/1").asString();

JsonPath jsonPath = new JsonPath(stringResponse);

jsonPath.делай, что хочеш

№ 3

Похоже, что обьект класса JsonPath можно создать из респонса.

JsonPath jsonPath = response.jsonPath();