

## **HOMEWORK-FormulaAPI INFO --> <http://ergast.com/mrd/methods/drivers/>**

- It's for historical formula one race information
- In this particular api , it decided to give you ml by default for response type and In this particular api , it decided to give you json if you add .json at the end of url

FOR EXAMPLE

- <http://ergast.com/api/f1/drivers.json> ---> return JSON

**BASE URL --> <http://ergast.com/api/f1/>**

### **TASK 1 :**

NOTE --> Solve same task with 4 different way

#### **- Use JSONPATH**

*int total = jsonpath.getInt("pathOfField")*

#### **- Use HAMCREST MATCHERS**

*then().body(.....)*

*Print givenName of Driver by using extract method after HamCrest*

#### **- Convert Driver information to Java Structure**

*Map<String,Object> driverMap=jsonpath.getMap("pathOfDriver")*

#### **- Convert Driver information POJO Class**

*Driver driver=getObject("pathOfDriver",Driver.class)*

- Given accept type is json
- And path param driverId is alonso
- When user send request /drivers/{driverId}.json
- Then verify status code is 200
- And content type is application/json; charset=utf-8 - And total is 1
- And givenName is Fernando
- And familyName is Alonso
- And nationality is Spanish

## TASK 2 :

NOTE --> Solve same task with 4 different way

### - Use JSONPATH

```
int total = jsonpath.getInt("pathOfField")
```

### - Use HAMCREST MATCHERS

```
then().body(.....)
```

*Print all names of constructor by using extract method after HamCrest*

### - Convert Constructor information to Java Structure

```
List<Map<String,Object>>
```

```
constructorMap=jsonpath.getList("pathOfConsts")
```

### - Convert Constructor information POJO Class

```
List<ConstructorPOJO>
```

```
constr=getObject("pathOfConstr",ConstructorPOJO.class)
```

**NOTE** —> There is a class in JAVA That's why give your class name *ConstrutorPOJO* to separate from existing one. Wrong imports may cause issue

- Given accept type is json
- When user send request /constructorStandings/1/constructors.json - Then verify status code is 200
- And content type is application/json; charset=utf-8
- And total is 17
- And limit is 30
- And each constructor has constructorId
- And constructor has name
- And size of constructor is 17 (using with hasSize)
- And constructor IDs has "benetton", "mercedes", "team\_lotus"
- And names are in same order by following list

```
List<String> names =Arrays.asList("Benetton", "Brabham-Repco", "Brawn", " BRM", "Cooper-Climax", "Ferrari", "Lotus-Climax", "Lotus-Ford", "Matra-Ford", " McLaren", "Mercedes", "Red Bull", "Renault", "Team Lotus", "Tyrrell", "Vanwall", " Williams");
```

GET <http://ergast.com/api/f1/drivers/:driverId.json>

Params • Authorization Headers (7) Body Pre-request Script Tests Settings

**Path Variables**

KEY	VALUE	DESCRIPTION
driverId	alonso	Description

Body Cookies Headers (9) Test Results Status: 200 OK Time: 163 ms Size: 75

Pretty Raw Preview Visualize JSON

```
1 {
2   "MRData": {
3     "xmlns": "http://ergast.com/mrd/1.5",
4     "series": "f1",
5     "url": "http://ergast.com/api/f1/drivers/alonso.json",
6     "limit": "30",
7     "offset": "0",
8     "total": "1",
9     "DriverTable": {
10      "driverId": "alonso",
11      "Drivers": [
12        {
13          "driverId": "alonso",
14          "permanentNumber": "14",
15          "code": "ALO",
16          "url": "http://en.wikipedia.org/wiki/Fernando_Alonso",
17          "givenName": "Fernando",
18          "familyName": "Alonso",
19          "dateOfBirth": "1981-07-29",
20          "nationality": "Spanish"
21        }
22      ]
23    }
24  }
25 }
```

## GET Single Driver

```
GET http://ergast.com/api/f1/constructorStandings/1/constructors.json

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Body Cookies Headers (9) Test Results Status: 200

Pretty Raw Preview Visualize JSON ↺

1 {
2   "MRData": {
3     "xmlns": "http://ergast.com/mrd/1.5",
4     "series": "f1",
5     "url": "http://ergast.com/api/f1/constructorstandings/1/constructors.json",
6     "limit": "30",
7     "offset": "0",
8     "total": "17",
9     "ConstructorTable": {
10      "constructorStandings": "1",
11      "Constructors": [
12        {
13          "constructorId": "benetton",
14          "url": "http://en.wikipedia.org/wiki/Benetton_Formula",
15          "name": "Benetton",
16          "nationality": "Italian"
17        },
18        {
19          "constructorId": "brabham-repco",
20          "url": "http://en.wikipedia.org/wiki/Brabham",
21          "name": "Brabham-Repco",
22          "nationality": "British"
23        },
24        {
```

## GET Constructors