# JSON

# All About serialization:

@Data – Lombok in general

@Builder --- to automatically build

e.g. 2 Classes:

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application, chat or text message

Description automatically generated

# All About deserialization:

## Way 1 :

De-serialization to POJO classes

## Way 2: dynamic (Java Streams way)

Java streams

Sample body -> there could be multiple objects inside value

{

{

"value": [

{

"File": {

"Name": "vitae\_quas-doloremque.txt"

},

"Id": 70,

"ID": 70,

"GUID": "2fc67167-bb63-4a8a-af23-2785b0403037"

}

]

}

It was Arraylist(Hashmap(String, anyType)))

Graphical user interface, text, application

Description automatically generated

# Way 3 : jayway JsonPath (cleanest dynamic approach)

<dependency>  
 <groupId>com.jayway.jsonpath</groupId>  
 <artifactId>json-path</artifactId>  
 <version>2.4.0</version>  
</dependency>

import com.jayway.jsonpath.JsonPath;  
import net.minidev.json.JSONArray;

JSONArray dataList = JsonPath.*parse*(response.getBody().asString())  
 .read(String.*format*("value[?(@.File.Name == '%s')].Id", fileName));  
*assertEquals*("Expected atleast 1 matching file, file actually present : " + dataList.size(), 1, dataList.size());  
return Integer.*parseInt*(dataList.get(0).toString());

To write this expression:

<https://jsonpath.herokuapp.com/>

Graphical user interface, application

Description automatically generated