

CSD-4464 Java EE

Json & Jackson



Json Datatype

- Acronym for Javascript object notation
- Denoted by the .json file extension type
- A way of describing Key:Value pairs
- Json objects are 0 or more Key:Value pairs wrapped in {}
- Keys are strings wrapped in double quotes
- Values can be a json object, double quote string, or an Array
- Key:values are separated by commas
- Arrays are denoted by [], and can contain strings, or objects or arrays

Json resources

- <https://jsonlint.com/> - great tool for validating json
- <http://www.json.org/> -good resource for understanding the jsonSpec

Json Example

```
{  
  "myKeyOne": "myValueOne",  
  "myKeyTwo": {  
    "nestObjectKey": ["string1", "string2"]  
  }  
}
```

Jackson ObjectMapper

- Jackson has been known as "the Java JSON library" or "the best JSON parser for Java". Or simply as "JSON for Java".
- <https://github.com/FasterXML/jackson>
- Utilized by many frameworks for handling JSON

<dependency>

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

 <artifactId>jackson-databind</artifactId>

 <version>2.12.1</version>

</dependency>

@JsonCreator @JsonProperty @Jacksonized ...

- Annotations used to tell Jackson how to deserialize json
- @JsonCreator tells it what constructor to use
- @JsonProperty tells it what json key corresponds to the parameter
- @Jacksonized tells Lombok to copy Jackson annotations to the builder
- @JsonAlias tells Jackson an alternative key to assign to the parameter

Given..

```
public class Student {  
    private String name;  
    private Integer age;  
  
    @JsonCreator  
    public Student(@JsonProperty("name") String name,  
                   @JsonProperty("age") Integer age) {  
        ....  
    }  
}
```

And..

```
ObjectMapper objectMapper = new ObjectMapper();  
Student student = new Student("joe", 22);  
objectMapper.writeValue(new File(" student.json"), student);
```

Will generate a json file containing

```
{  
  "name" : "joe",  
  "age" : 22  
}
```

note* objectMapper.writeValueAsString(student); also generates a string of the above json

Also..

```
Student student = objectMapper.readValue(new File("student.json"),  
Student.class);
```

```
student.getName(); //joe
```

Using Collections

```
List<Student> listStudent = objectMapper.readValue(students.json, new  
TypeReference<List<Student>>(){});
```