

How To Convert Java Object To / From JSON (Jackson)



Posted on August 9, 2011

By [mkyong](#)

 Like 9.2k

Follow [@mkyong](#)

Get the latest in your inbox for free.

Subscribe

Convert Java To/From JSON Gson: free open-source JSON library for Java with an easy to use API code.google.com

Jackson is a High-performance JSON processor Java library. In this tutorial, we show you how to use Jackson's data binding to convert Java object to / from JSON.

For object/json conversion, you need to know following two methods :

```
//1. Convert Java object to JSON format
ObjectMapper mapper = new ObjectMapper();
mapper.writeValue(new File("c:\\user.json"), user);
```

```
//2. Convert JSON to Java object
ObjectMapper mapper = new ObjectMapper();
User user = mapper.readValue(new File("c:\\user.json"), User.class);
```

Note

Both `writeValue()` and `readValue()` has many overloaded methods to support different type of inputs and outputs. Make sure check it out.

000

1. Jackson Dependency

Jackson contains 6 separate jars for different purpose, check [here](#). In this case, you only need “jackson-mapper-asl” to handle the conversion, just declares following dependency in your `pom.xml`

```
<repositories>
  <repository>
    <id>codehaus</id>
    <url>http://repository.codehaus.org/org/codehaus</url>
  </repository>
</repositories>

<dependencies>
  <dependency>
    <groupId>org.codehaus.jackson</groupId>
    <artifactId>jackson-mapper-asl</artifactId>
    <version>1.8.5</version>
  </dependency>
</dependencies>
```

For non-maven user, just get the [Jackson library here](#).

[RT Java Virtual Machine](#)

Aonix Perc VM for RealTime Systems. Downloads-DataSheets & White Papers
www.Atego.com

2. POJO

An user object, initialized with some values. Later use Jackson to convert this object to / from JSON.

```
package com.mkyong.core;
```

```

import java.util.ArrayList;
import java.util.List;

public class User {

    private int age = 29;
    private String name = "mkyong";
    private List<String> messages = new ArrayList<String>() {
        {
            add("msg 1");
            add("msg 2");
            add("msg 3");
        }
    };

    //getter and setter methods

    @Override
    public String toString() {
        return "User [age=" + age + ", name=" + name + ", " +
            "messages=" + messages + "]\n";
    }
}

```

3. Java Object to JSON

Convert an “user” object into JSON formatted string, and save it into a file “**user.json**”.

```

package com.mkyong.core;

import java.io.File;
import java.io.IOException;
import org.codehaus.jackson.JsonGenerationException;
import org.codehaus.jackson.map.JsonMappingException;
import org.codehaus.jackson.map.ObjectMapper;

```

```
public class JacksonExample {
    public static void main(String[] args) {

        User user = new User();
        ObjectMapper mapper = new ObjectMapper();

        try {

            // convert user object to json string, and save to a file
            mapper.writeValue(new File("c:\\user.json"), user);

            // display to console
            System.out.println(mapper.writeValueAsString(user));

        } catch (JsonGenerationException e) {

            e.printStackTrace();

        } catch (JsonMappingException e) {

            e.printStackTrace();

        } catch (IOException e) {

            e.printStackTrace();

        }

    }
}
```

Output

```
{"age":29,"messages":["msg 1","msg 2","msg 3"],"name":"mkyong"}
```

Note

Above JSON output is hard to read. You can enhance it by enable the [pretty print feature](#).

4. JSON to Java Object

Read JSON string from file “**user.json**“, and convert it back to Java object.

```
package com.mkyong.core;

import java.io.File;
import java.io.IOException;
import org.codehaus.jackson.JsonGenerationException;
import org.codehaus.jackson.map.JsonMappingException;
import org.codehaus.jackson.map.ObjectMapper;

public class JacksonExample {
    public static void main(String[] args) {

        ObjectMapper mapper = new ObjectMapper();

        try {

            // read from file, convert it to user class
            User user = mapper.readValue(new File("c:\\user.json"), User.class);

            // display to console
            System.out.println(user);

        } catch (JsonGenerationException e) {

            e.printStackTrace();

        } catch (JsonMappingException e) {

            e.printStackTrace();

        } catch (IOException e) {
```

```
e.printStackTrace();

    }

}

}
```

```
User [age=29, name=mkyong, messages=[msg 1, msg 2, msg 3]]
```

References

- 1. [Download Jackson](#)
- 2. [Jackson Official Site](#)
- 3. [Gson example – convert Java object to / from JSON](#)

Tags : [convert](#) [jackson](#) [java](#) [json](#)



[mkyong](#)

Founder of Mkyong.com, love Java and open source stuffs. Follow him on [Twitter](#), or befriend him on [Facebook](#) or [Google Plus](#).



Related Posts

- ▶ [Java JSON Tutorial](#)
- ▶ [How to convert Java Map to / from JSON \(Jackson\)](#)
- ▶ [How to enable pretty print JSON output \(Jackson\)](#)
- ▶ [JSON.simple example – Read and write JSON](#)
- ▶ [Jackson Streaming API to read and write JSON](#)

Popular Posts

- ▶ [Top 8 Java People You Should Know](#)
- ▶ [Top 20 Java Websites](#)
- ▶ [Top 5 Free Java eBooks](#)
- ▶ [Top 10 Java Regular Expression Examples](#)
- ▶ [Top 5 Open Source Q&A Systems](#)

You might also like following tutorials :



[Leonardo](#)

November 4, 2012 at 5:11 am

When I use:

```
objectMapper.readValue(json, targetClass);
```

And there is some characters with punctuation like “Ñão” readValue transform to something like: “NÃ&O”

I thing the throuble is about UTF8 decode. Can you help-me?

Reply



murkein

August 10, 2012 at 6:55 pm

Excellent :)

Reply



shankar

July 31, 2012 at 8:53 am

Is there any way to do WITHOUT FILE, writing to and reading from a file is very expensive.

Reply



Anuj

June 17, 2012 at 4:42 pm

Hi,

I wrote REST Service using CXF and Spring. Service is returning output in xml when using MediaType.APPLICATION_XML but i want to return output as JSON and not XML. I found that instead of just using @Produces({ @application/json }), there are some JSONProvider needs to be used in Spring Configuration. I don't

want to use CXF Jettison but needs to use "Jackson". can you please assist on below?

Ex.

@GET

@Produces({ MediaType.APPLICATION_JSON})

public Customer getAllCustomers() {

return customers; //customers is Array list of Customer

}

Reply



Yogesh

March 30, 2012 at 12:42 am

Hi

I have some string variables and I need to create JSON object from them and return JSON Object from my function. I need to use jackson json library for project requirements. Can you please tell how to return a json object and not write to a file.

```
Map<String, String> resultData = new HashMap<String, String>();
    resultData.put("id", id);
    resultData.put("Name", Name);
    resultData.put("dob", dob);
    resultData.put("gender", gender);

    ObjectMapper mapper = new ObjectMapper();
    mapper.writeValueAsString(writeData);
```

Reply



Sidhardh

January 23, 2012 at 1:51 pm

org.codehaus.jackson.map.JsonMappingException: No serializer found for class edu.learning.json.json_examples.User and no properties discovered to create BeanSerializer (to avoid exception, disable SerializationConfig.Feature.FAIL_ON_EMPTY_BEANS))

I'm getting the above exception while executing objectMapper.writeValue. How to solve this

Reply



Faisal

June 20, 2012 at 12:02 am

Although I am new with JSON but may be the following findings will help.

I was having the same issue and then resolved that by adding JSON Annotation with the POJO called "User.java". You need to add "@JsonAutoDetect" just in front of your "public class User{..." declaration and "@JsonProperty" in front of any attributes (here it is: private int age = 29; and private String name = "mkyong";) and methods (here it is: private List messages = new ArrayList() {...}).

Note that what ever element/information you wanna pass to your main Java Class (here it is "public class JacksonExample{...}") from the POJO, add the proper JSON Annotation.

Reply



Pulkit Singhal

August 24, 2012 at 2:22 am

You actually need to disable the empty bean failure behaviour. It is not exactly simple to pass Jackson configuration through Jersey but its relatively easy after 15~20 mins of trying.

This link:

<http://jersey.java.net/nonav/documentation/latest/json.html>

talks about configuring Jackson for Jersey:

“Jackson JSON processor could be further controlled via providing custom Jackson ObjectMapper instance. This could be handy if you need to redefine the default Jackson behaviour and to fine-tune how your JSON data structures look like. Detailed description of all Jackson features is out of scope of this guide. The example below gives you a hint on how to wire your ObjectMapper instance into your Jersey application.”

Download <https://maven.java.net/service/local/artifact/maven/redirect?r=releases&g=com.sun.jersey.samples&a=jacksonjsonprovider&v=1.13&c=project&e=zip> to get a complete example using POJO based JSON support.

Reply



C

December 21, 2011 at 3:45 am

To make this work with the latest (1.9.3) Jackson, there must be getters and setters for the members in the User class.

Reply

I'd love your thoughts

Your email address will not be published. Required fields are marked *

<input type="text"/>	Name *
<input type="text"/>	Email *
<input type="text"/>	Website

Comment

Note

To post source code in comment, uses `<pre lang="language"></pre>` tag, for examples :

- > `<pre lang="java"> Java source code here </pre>`
- > `<pre lang="xml"> XML here </pre>`

Post Comment

☐ Notify me of followup comments via e-mail

All Available Tutorials

Java Core Technologies :
Java I/O, Java RegEx, Java XML, Java JSON, JDBC, Java Misc
J2EE Frameworks :
Hibernate, JSF 2.0, Spring Core, Spring MVC, Spring Security, Apache Wicket, Struts 1.x, Struts 2.x
Web Service :
JAX-WS (SOAP), JAX-RS (REST)
Build Tools :
Maven, Archiva
Unit Test Frameworks :
JUnit, TestNG

Favorites Links

Android Developer
Google App Engine using Java
DZone - Fresh Links
Official Java EE 5 Tutorial
Official Java EE 6 Tutorial
Spring 2.5.x documentation
Spring 3.1.x documentation
Hibernate core documentation
Java SE 6.0 API documentation
Java EE 6.0 API documentation
Java Secure Socket Extension (JSSE) Reference Guide
JSR home page

Friends & Links

Java Training
JavaScript Training
Java Code Geeks
PHP Tutorials
TenthOfMarch
Find Free Icons
Web Security Blog
Web Development

About Us

Mkyong.com is a weblog dedicated to Java/J2EE developers and Web Developers. We constantly publish useful tricks, tutorials on J2EE or web development.

All examples are simple, easy to read, and full source code available, and of course well tested in our development environment.

We're Social

- 1. Twitter - Follow Me
- 2. Facebook - Like Me
- 3. Google Plus - Add Me
- 4. RSS - Subscribe Me

Others :
Android, Google App Engine, jQuery, Java
MongoDB, Quartz Scheduler

JSF home page
JSF home page
Eclipse IDE for Java developer
Struts 1.3 documentation
Struts 2.2 documentation
Maven home page
Maven central repository Search
Java.Net Maven repository
Ant home page
JAX-WS Official Website
JAX-RS Official Website (Jersey)
MongoDB Official Website

Copyright © 2008-2012 Mkyong.com, all rights reserved. Web Hosting Powered by [Liquid Web](#)