O'REILLY
Answers
Clever hacks. Creative ideas.
Innovative solutions.

Your Account: **Sign In**

Search

Shopping Cart

| Home | Shop | News & Commentary | Answers | Safari Books Online | Conferences | School of Technology | Community |

Trending Topics: windows 8 | makefile | xml | json | jQuery | html | ebook | windows | More...

How to Parse JSON in Java

Submitted by tmo9d

Posted Sep 23 2009 12:25 PM

+ 3

Use json-lib, a library which adds JSON support to any Java program. json-lib can take a String and turn it into a JSONObject which can then be used to retrieve specific attributes.

1. Add this dependency to your project:

```xml
<dependency>
    <groupId>net.sf.json-lib</groupId>
    <artifactId>json-lib</artifactId>
    <version>2.3</version>
    <scope>compile</scope>
</dependency>
```

What does this mean? See How to Add a Dependency to a Java Project

2. Put the following JSON sample in your classpath:

```json
{'foo':'bar',
 'coolness':2.0,
 'altitude':39000,
 'pilot':{'firstName':'Buzz',
          'lastName':'Aldrin'},
```

```
    'mission':'apollo 11'}
```

3. Load the resource from the classpath and parse this JSON as follows:

```java
package com.discursive.answers;

import java.io.InputStream;

import net.sf.json.JSONObject;
import net.sf.json.JSONSerializer;

import org.apache.commons.io.IOUtils;

public class JsonParsing {

    public static void main(String[] args) throws Exception {
        InputStream is =
                JsonParsing.class.getResourceAsStream( "sample-json.txt");
        String jsonTxt = IOUtils.toString( is );

        JSONObject json = (JSONObject) JSONSerializer.toJSON( jsonTxt );
        double coolness = json.getDouble( "coolness" );
        int altitude = json.getInt( "altitude" );
        JSONObject pilot = json.getJSONObject("pilot");
        String firstName = pilot.getString("firstName");
        String lastName = pilot.getString("lastName");

        System.out.println( "Coolness: " + coolness );
        System.out.println( "Altitude: " + altitude );
        System.out.println( "Pilot: " + lastName );
    }
}
```

Note that JSONSerializer returns a JSON object. This is a general object which could be a JSONObject or a JSONArray depending on the JSON you are trying to parse. In this example, since I know that the JSON is a JSONObject, I can cast
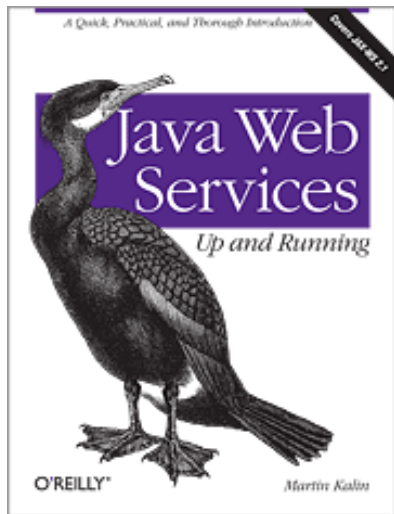
the result directly to a JSONObject. If you are dealing with JSON that could return a JSONArray, you'll likely want to check the type of the object that is returned by toJSON.

This sample project is available on GitHub here: http://github.com/to...le-json-parsing

For more information about the json-lib project, see JSON-LIB project page

Learn more about this topic from **Java Web Services: Up and Running**.

This quick, practical, and thorough introduction to Java web services -- the JAX-WS and JAX-RS APIs -- offers a mix of architectural overview, complete working code examples, and short yet precise instructions for compiling, deploying, and executing a sample application. You'll not only learn how to write web services from scratch, but also how to integrate existing services into your Java applications.

See what you'll learn

Tags: java   json

0 Subscribe   Email   Tweet ‹5   +1 ‹4   Like ‹26

3 Replies

Reply

Alternative solution by Stuart12345
Posted Sep 30 2011 10:13 AM

👍 0 👎

Be sure to add the classifier:

<dependency>
<groupId>net.sf.json-lib</groupId>
<artifactId>json-lib</artifactId>
<version>2.4</version>
<classifier>jdk15</classifier>
</dependency>

see http://stackoverflow...net-sf-json-lib

Alternative solution by Adam Musial-Bright 🖼

Posted Oct 03 2011 09:39 AM

Gson, the Google JSON library is also very nice.

You will need a simple POJOs for the JSON representation:

```
// JSON representation
class MyJson {

  public MyJson() {}

  private String foo;
  private float coolness;
  // other attributes

  public void setFoo(String foo) {
    this.foo = foo;
  }

  public String getFoo() {
    return foo;
  }
}
```

```java
    public void setCoolness(float coolness) {
        this.coolness = coolness;
    }


    public float getCoolness() {
        return coolness;
    }


}
```

Now you can convert a JSON representation into a JSON object:

```java
Gson gson = new Gson();
MyJson myJson = new gson.fromJson(jsonTxt, MyJson.class);


// now you have a real java object
System.out.println(myJson.getFoo());
```

done.

Alternative solution by handark
Posted Mar 30 2012 07:21 AM

👍 0 👎

Gson is a very good library.

Thanks Adam

O'REILLY®

About O'Reilly

Academic Solutions

Jobs

Contacts

Corporate Information

Press Room

Privacy Policy

Terms of Service

Writing for O'Reilly

Community

Authors

Community & Featured Users

Forums

Membership

Newsletters

O'Reilly Answers

RSS Feeds

User Groups

More O'Reilly Sites

igniteshow.com

makerfaire.com

makezine.com

craftzine.com

labs.oreilly.com

Partner Sites

PayPal Developer Zone

O'Reilly Insights on Forbes.com

Shop O'Reilly

Customer Service

Contact Us

Shipping Information

Ordering & Payment

The O'Reilly Guarantee