JSON-B: Java™ API for JSON Binding

Version 1.0 Early Draft
January 30, 2015

Editors:
Martin Grebac

Comments to: users@jsonb-spec.java.net

Oracle Corporation
500 Oracle Parkway, Redwood Shores, CA 94065 USA.

JSR-367 Java API for JSON Binding ("Specification")
Version: 1.0
Status: Early Draft
Release: January 30, 2015

TBD

Contents

Chapter 1

Introduction

This specification defines binding API between Java objects and JSON [1] documents. Readers are assumed to be familiar with JSON; for more information about JSON, see:

- Architectural Styles and the Design of Network-based Software Architectures[2]
- The REST Wiki[3]
- JSON on Wikipedia[4]

1.1 Status

This is an early draft; this specification is not yet complete. A list of open issues can be found at:

http://java.net/jira/browse/JSONB_SPEC

The corresponding Javadocs can be found online at:

http://jsonb-spec.java.net/

The reference implementation will be obtained from:

http://eclipselink.org/

The expert group seeks feedback from the community on any aspect of this specification, please send comments to:

users@jsonb-spec.java.net

1.2 Goals

The following are the goals of the API:

JSON Support binding (marshalling and unmarshalling) for all RFC 7159 compatible JSON documents.

Relationships to JSON Related specifications JSON related specifications will be surveyed to determine their relationship to JSON-Binding.

Consistency Maintain consistency with JAXB (Java API for XML Binding) and other JavaEE and SE APIs.

Convention Define default mapping of Java classes and instances to JSON document counterparts.

Customization Allow to customize the default mapping definition.

Ease Of Use Default use of the APIs SHOULD NOT require prior knowledge of the JSON document format and specification.

Partial Mapping In many usecases, only a subset of JSON Document is required to be mapped to a Java object instance.

Integration Define or enable integration with following Java EE technology standards: JSR 353 - Java API for JSON Processing 1.1 JSR 349, ... - Bean Validation (BV) 1.1 (2.0) JSR 370 - JavaTM API for RESTful Web Services (JAX-RS) 2.1

1.3 Non-Goals

The following are non-goals:

Preserving equivalence (Round-trip) The specification recommends, but does not require equivalence of content for unmarshalled and marshalled JSON documents.

JSON Schema Generation of JSON Schema from Java classes, as well as validation based on JSON schema is out of scope of this specification.

JEP 148 Lightweight JSON API Support Support and integration with Lightweight JSON API as defined within JEP 198 is out of scope of this specification. Will be reconsidered in future specification revisions.

1.4 Conventions

The keywords 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL', 'SHALL NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED', 'MAY', and 'OPTIONAL' in this document are to be interpreted as described in RFC 2119[**?**].

Java code and sample data fragments are formatted as shown in figure 1.1:

URIs of the general form 'http://example.org/...' and 'http://example.com/...' represent application or context-dependent URIs.

All parts of this specification are normative, with the exception of examples, notes and sections explicitly marked as 'Non-Normative'. Non-normative notes are formatted as shown below.

Note: *This is a note.*

Figure 1.1: Example Java Code

```
1  package com.example.hello;
2
3  public class Hello {
4      public static void main(String args[]) {
5          System.out.println("Hello World");
6      }
7  }
```

1.5 Terminology

Databinding Process which defines representation of information in an JSON document as an object instance, and vice versa.

Unmarshalling Process of reading a JSON document and construction a tree of content objects, where each object corresponds to part of JSON document, thus the content tree reflects the document's content.

Marshalling Inverse process to unmarshalling. Process of traversing content object tree and writing a JSON document that reflects the tree's content.

1.6 Expert Group Members

This specification is being developed as part of JSR 367 under the Java Community Process. It is the result of the collaborative work of the members of the JSR 367 Expert Group. The following are the present expert group members:

- Martin Grebac (Oracle)
- Martin Vojtek (Oracle)
- Hendrik Saly (Individual Member)
- Gregor Zurowski (Individual Member)
- Inderjeet Singh (Individual Member)
- Eugen Cepoi (Individual Member)
- Przemyslaw Bielicki (Individual Member)
- Kyung Koo Yoon (TmaxSoft, Inc.)
- Otavio Santana (Individual Member)
- Rick Curtis (IBM)
- Alexander Salvanos (Individual Member)

1.7 Acknowledgements

During the course of this JSR we received many excellent suggestions. Special thanks to ... .

During the course of this JSR we received many excellent suggestions on the JSR java.net project mailing lists, thanks in particular to ... for their contributions. The following individuals have also made invaluable technical contributions: ... .

Chapter 2

Runtime API

JSON-B Runtime API provides access to marshalling and unmarshalling operations for manipulating with JSON documents and mapped JSON-B classes and instances. The full specification of the binding framework is available in the javadoc for the `javax.json.bind package`.

Chapter 3

Default Mapping

This section defines the default binding mapping (representation) of Java components and classes within Java programming language to JSON documents. The default mapping defined here can be further customized as specified in Chapter 4 - Customizing Mapping.

3.1 Names and identifiers

3.2 Java Package

3.3 Java Class

Chapter 4

Customizing Mapping

JSON-B TBD

Bibliography

[1] Ed. T. Bray. The javascript object notation (json) data interchange format. RFC 2070-1721, IETF, March 2014.

[2] R. Fielding. Architectural Styles and the Design of Network-based Software Architectures. Ph.d dissertation, University of California, Irvine, 2000. See http://roy.gbiv.com/pubs/dissertation/top.htm.

[3] REST Wiki. Web site. See http://rest.blueoxen.net/cgi-bin/wiki.pl.

[4] Technical report, Wikipedia, November 2014.