




ORACLE[®]

JSR-353 : Java API for Processing JSON

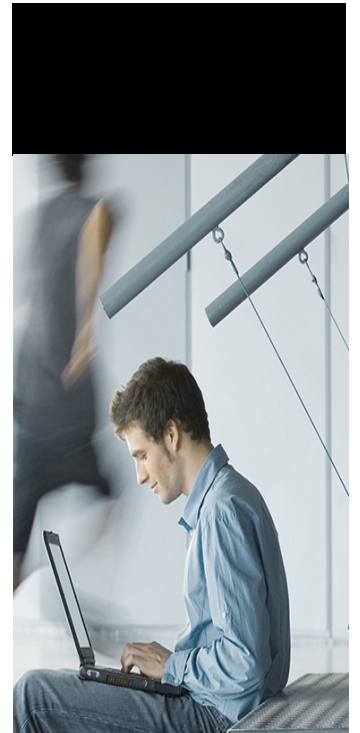
Jitendra Kotamraju



The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Agenda

- Overview
- JAX-RS Usage
- Standardization





Overview

JSON

- JSON is a light-weight data exchange format
 - Easy for humans/machines to read and write
 - For e.g.:

```
{"name": "Bob", "age": 20, "phone": ["276 1234", "1234567"]}
```
- JSON is used by popular web sites in their RESTful web services
 - Facebook, Twitter, Amazon, ...
 - Twitter Streaming API discontinues XML



Overview

JSON usages

- Policy in Amazon SQS

```
{  
  ...  
  "Statement": {  
    "Effect": "Allow",  
    "Principal": { "AWS": "123456789012" },  
    "Action": "sqs:SendMessage",  
    "Resource": "/987654321098/queue1"  
  }  
}
```



Overview

JSON usages

- Followers in Twitter API

```
{  
  "previous_cursor": 0,  
  "ids": [143206502, 143201767, 777925],  
  "previous_cursor_str": "0",  
  ...  
}
```



JAX-RS

XML Usage

- JAX-RS applications handle XML using JAXP API

```
@Produces("application/xml")
public Source getBook(String id) {
    return new StreamSource(...);
}
```



JAX-RS

XML Usage

- JAX-RS applications handle XML using JAXB API

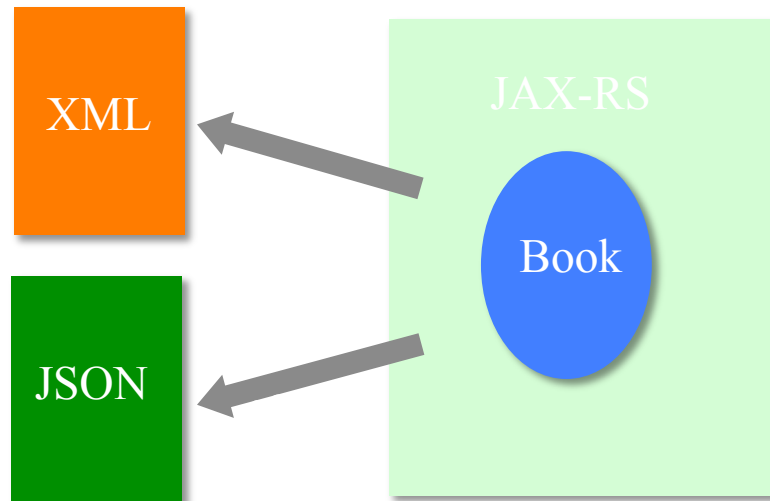
```
@Produces("application/xml")  
public Book getBook(String id) {  
    return new Book(...);  
}
```


JAX-RS

DataBinding

- JAX-RS content negotiation

```
@Produces({ "application/xml", "application/json" })  
public Book getBook(String id) {  
    return new Book();  
}
```





JAX-RS

JSON Solutions & Limitations

- A custom `MessageBodyWriter` that converts to JSON
 - `JSONObject` (For e.g. `json.org`'s API) → JSON
 - JAXB → StAX → JSON (For e.g. using `jettison`)
 - POJO/JAXB → JSON (For e.g. using `jackson`, `eclipseLink` etc.)
- No standard API
- Some solutions have technical limitations
- Applications/Frameworks need to bundle the libraries



Standard API

Advantages

- Application can use standard types
- Leaner, portable applications



Standard API

Contents

- Parsing/Processing JSON
- Data binding : JSON text <-> Java Objects
- Two JSRs: Processing/Parsing (JSON-P), Binding (JSON-B)
 - Similar to JAXP and JAXB
 - Close collaboration between the two

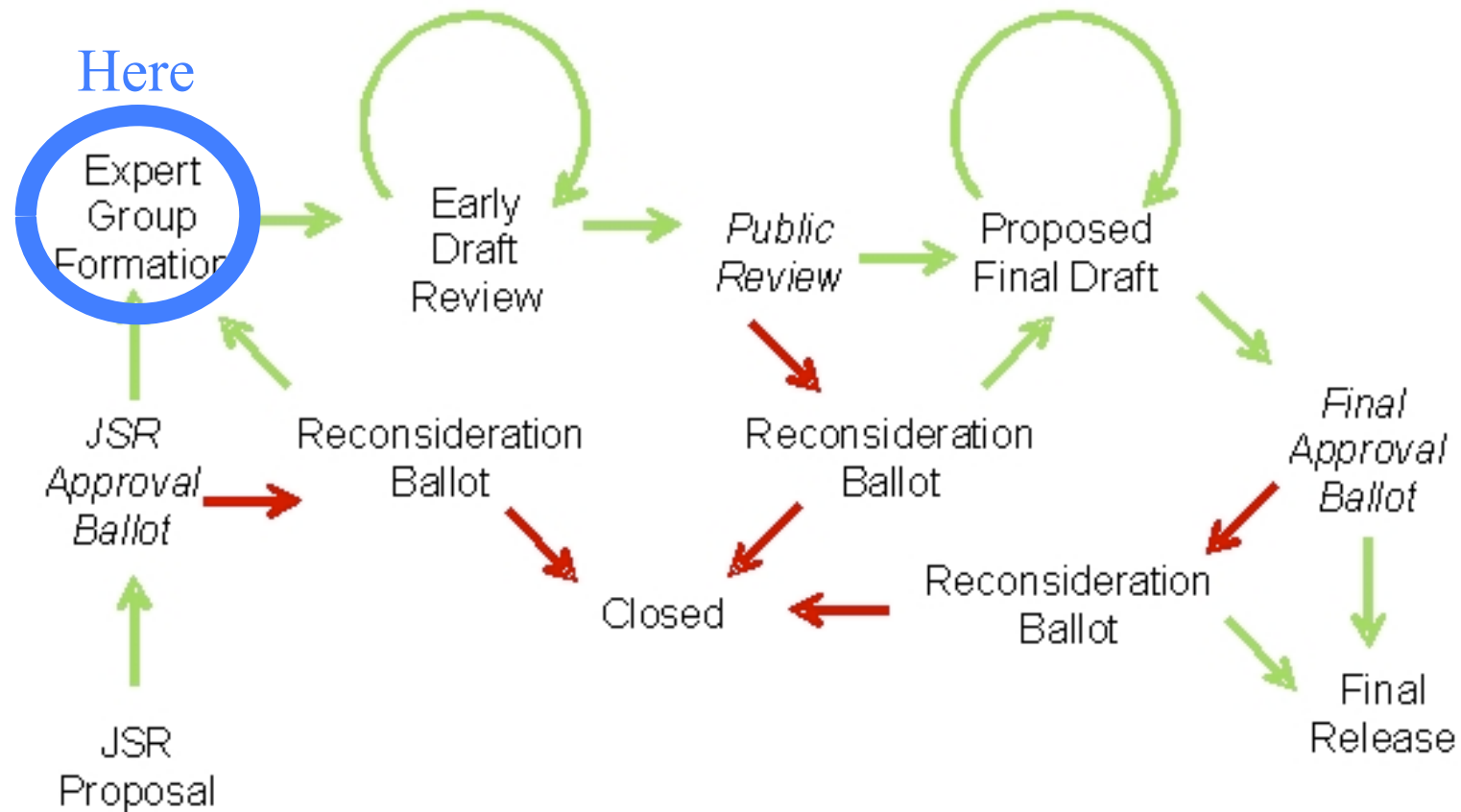


Java API for Processing JSON (JSON-P)

JSR-353

- Streaming API to produce/consume JSON
 - Similar to StAX API in XML world
- Object model API to represent JSON
 - Similar to DOM API in XML world
- Align with Java EE 7 schedules
- JSR Supporters
 - fasterxml.com(Jackson), [Doug Crockford](http://doug.crockford.com)(json.org)

JSR-353 : Status





JSR-353 Transparency

Open Source Project

- json-processing-spec java.net open source project is used for JSR-353
- Mailing lists:
 - users@json-processing-spec.java.net
 - jsr353-experts@json-processing-spec.java.net
- Issue Tracker:
 - http://java.net/jira/browse/JSON_PROCESSING_SPEC



Resources

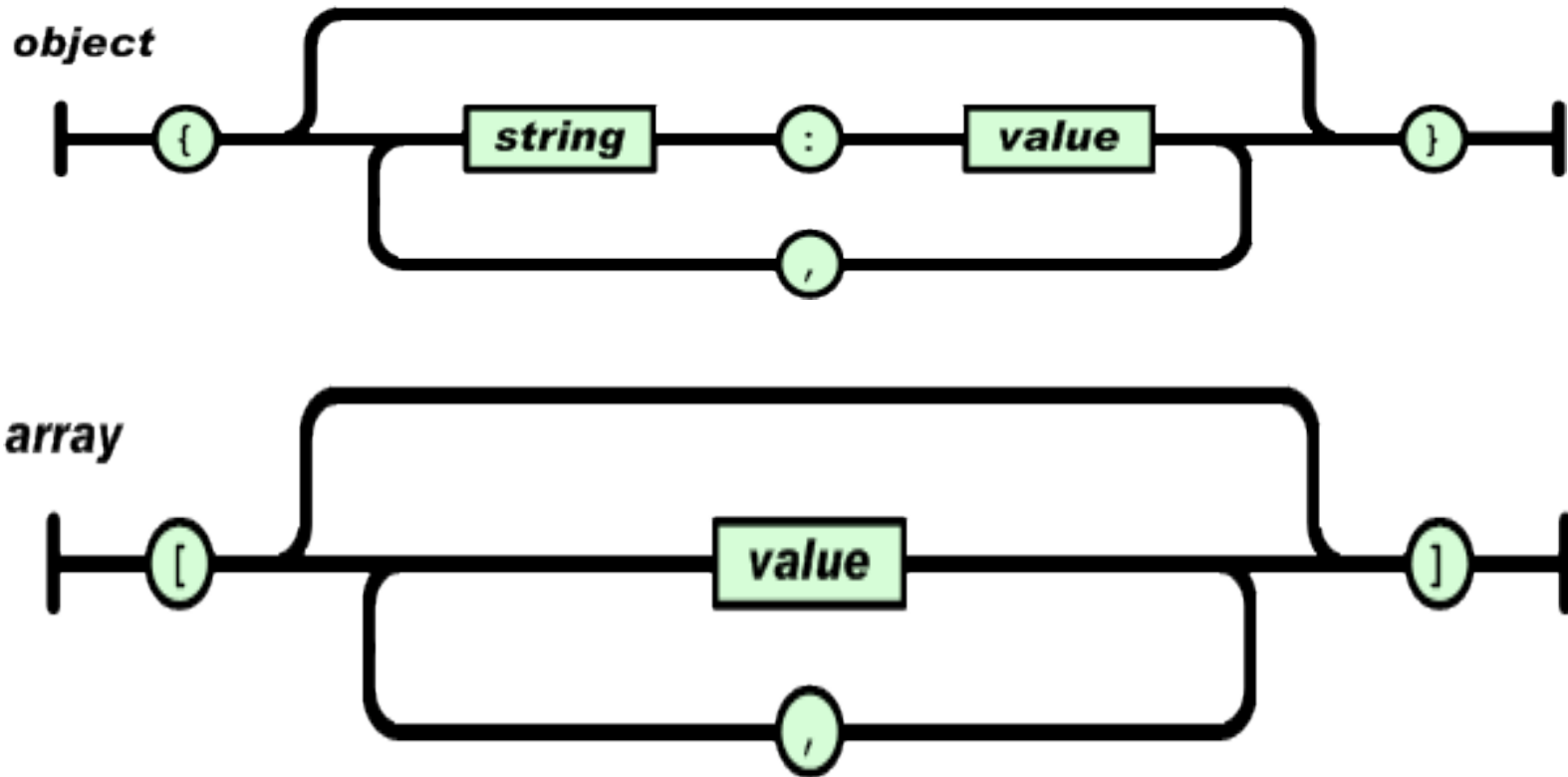
- <http://json-processing-spec.java.net>

Q&A



Parsing API

JSON Grammar

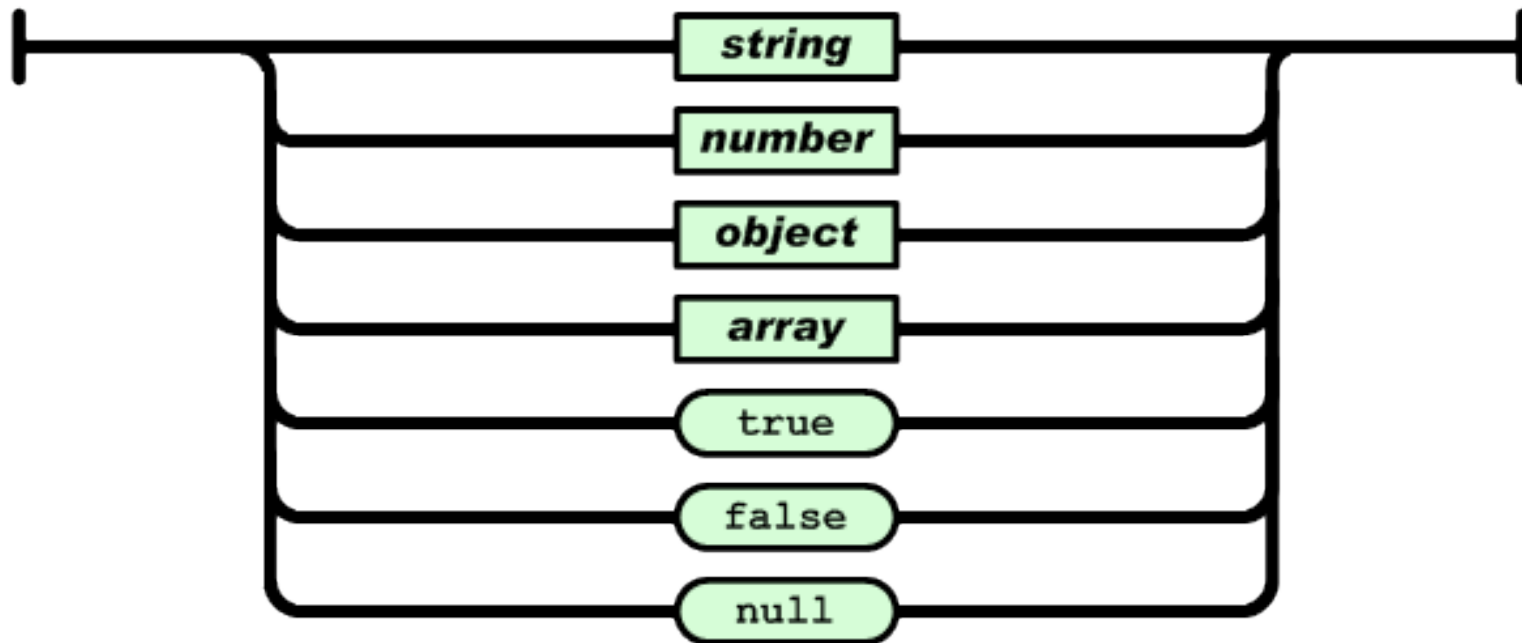




Parsing API

JSON Grammar

value





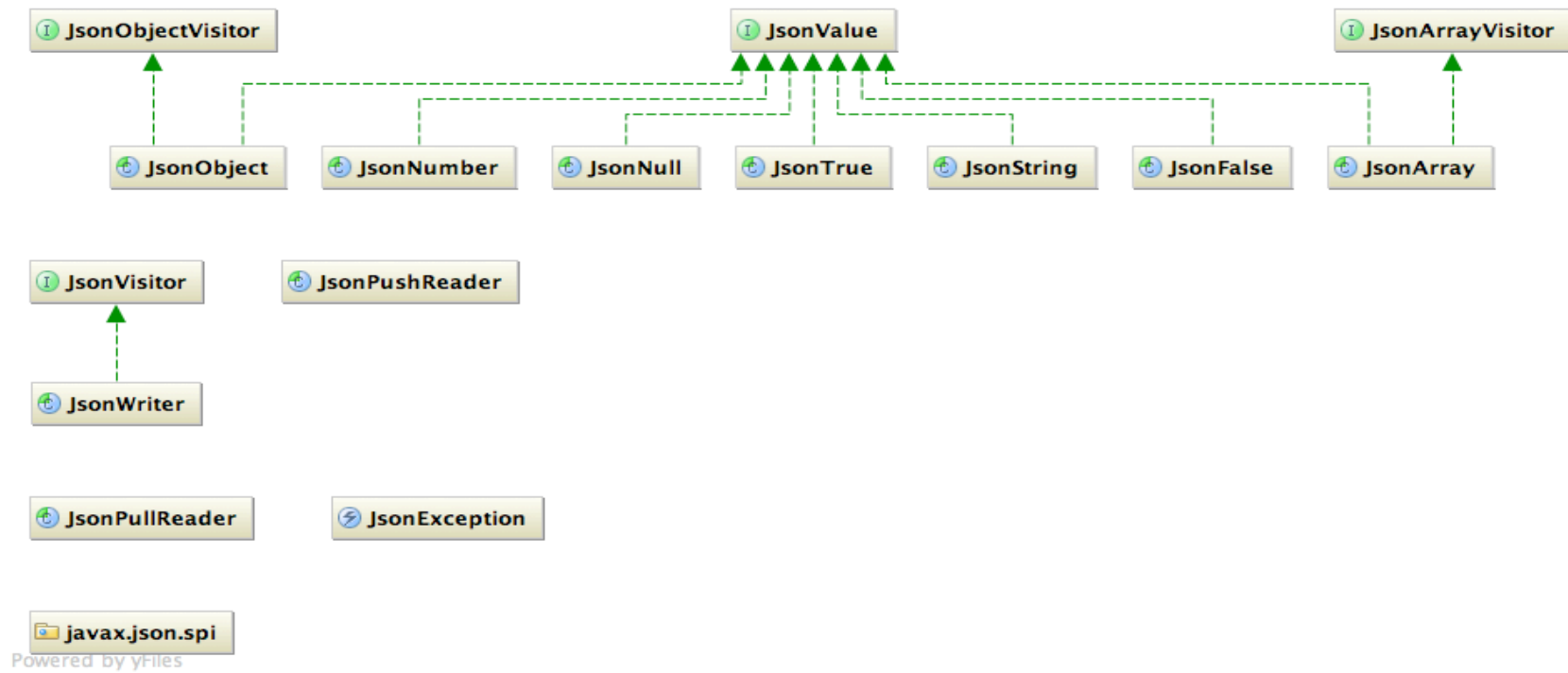
JSR-353 API

Processing API

- API initial proposal to EG
 - Based visitor pattern (similar to ASM, JSR 269 API, ...)
 - Works nicely with streaming and tree API
 - Providers plug-in their implementations

JSR-353 API

UML class diagram



ORACLE®