Paul Romer

CSD420

Module 11 Assignment

February 28, 2025

Jackson – 'the Java JSON library'

Jackson is a widely used Java-based open-source library for processing JSON data. It is known for its efficiency, versatility, and ease of use, making it a preferred choice for many Java developers, often used in the Spring framework. Developed by FasterXML, Jackson provides a suite of features that enable reading, writing, and transforming JSON data with minimal configuration in a better way than the built in Java parser (Evans). The API supports various data-binding mechanisms, high-performance streaming, and extensive customization, which makes it ideal for a variety of applications, including web services, data transformation, and persistence (FasterXML).

Jackson was developed by Tatu Saloranta in 2007 while working at Amazon. While working with XML as an underlying data format across web services. He read about JSON and quickly saw how it would allow simple conversion between JSON and JAVA objects. Within a month, he wrote a simple streaming parser and generator for JSON that was the beginning of what became Jackson (Saloranta).

Initially, Jackson was designed as a high-performance streaming JSON parser, with an emphasis on efficiency and modularity. Tatu assumed that if it gained traction, someone else would build robust data-binding capabilities. After the initial release in 2007, Jackson did get relative publicity, but was used as is. No one stepped up to build the data binding layer. The popularity of Jackson led him to extend Jackson himself - beyond simple streaming, introducing data-binding functionality to seamlessly map JSON to Java objects like maps and lists. By late 2008, he released the first version of Jackson with serialization support, followed shortly by deserialization in early 2009 (Saloranta).

As Jackson gained traction, developers recognized its speed and flexibility compared to alternatives like org.json. In time, the library evolved to include additional features, such as support for polymorphic type handling, mix-in annotations, and integration with widely used Java frameworks like Spring. Over time, Jackson expanded beyond JSON, adding modules for handling XML, YAML, and CSV, further solidifying its position as a comprehensive data processing tool. Today, it remains one of the most popular JSON parsers for Java, known for its performance, extensibility, and seamless integration with modern applications (Saloranta).

Jackson can be use in a Java project by including the Jar files as dependencies. They can be downloaded at https://mvnrepository.com/artifact/com.fasterxml.jackson.core/jackson-core/2.18.2 . If used with a build tool like Maven, it should be included as a dependency in the pom.xml file (Maven).

```
<!-- https://mvnrepository.com/artifact/com.fasterxml.jackson.core/jackson-core -->
<dependency>
    <groupId>com.fasterxml.jackson.core</groupId>
    <artifactId>jackson-core</artifactId>
    <version>2.18.3</version>
</dependency>
```

Jackson has three primary modules, jackson-core for streaming, jackson-anotations for annotations, and jackson-databind for databinding. These are the three main modules, while there are others that enable more features (Evans).

The example Java program demonstrates how Jackson facilitates JSON serialization and deserialization in Java. The JSON structure includes a worldChampion field, which holds the name of the current championship team ("McLaren"), and a teams array that lists a few of the Formula One teams. The FormulaOne class represents this JSON structure in Java, with corresponding fields and getter/setter methods. Using Jackson's ObjectMapper, the program first serializes the FormulaOne object into a JSON string, showing how Java objects can be easily converted into JSON format. It then deserializes the JSON back into a FormulaOne object, demonstrating how Jackson seamlessly maps JSON data to Java objects – like a String and a List. This feature makes Jackson a powerful tool for handling structured data in applications that interact with APIs.

```
JSON
{
    "worldChampion": "McLaren",
    "teams": ["Red Bull", "Mercedes", "Ferrari", "McLaren", "Aston Martin"]
}
```

```
import com.fasterxml.jackson.databind.ObjectMapper;
import java.util.Arrays;
import java.util.List;

class FormulaOne {
    private String worldChampion;
    private List<String> teams;

    // Default constructor (required for Jackson)
    public FormulaOne() {}
    // Constructor
    public FormulaOne(String worldChampion, List<String> teams) {
        this.worldChampion = worldChampion;
        this.teams = teams;
    }
    // Getters and setters
    public String getWorldChampion() {
        return worldChampion;
    }
    public void setWorldChampion(String worldChampion) {
        this.worldChampion = worldChampion;
    }
    public List<String> getTeams() {
        return teams;
    }
    public void setTeams(List<String> teams) {
        this.teams = teams;
    }
// Continued on Next Page - Main Method
```

```
public static void main(String[] args) throws Exception {

        ObjectMapper objectMapper = new ObjectMapper();


        // Creating a FormulaOne object

        FormulaOne f1 = new FormulaOne("McLaren", Arrays.asList("Red Bull",
"Mercedes", "Ferrari", "McLaren", "Aston Martin"));


        // Converting Java Object to JSON

        String jsonString = objectMapper.writeValueAsString(f1);

        System.out.println("Serialized JSON:\n" + jsonString);


        // Converting JSON to Java Object

        FormulaOne deserializedF1 = objectMapper.readValue(jsonString,
FormulaOne.class);

        System.out.println("Deserialized Object:\nWorld Champion: " +
deserializedF1.getWorldChampion());

    }

}
```

Link to JAR download:
https://mvnrepository.com/artifact/com.fasterxml.jackson.core/jackson-core/2.18.2

Actual JAR Link: https://repo1.maven.org/maven2/com/fasterxml/jackson/core/jackson-core/2.18.2/jackson-core-2.18.2.jar

Sources

Evans, Ben. “Efficient JSON Serialization with Jackson and Java.” Oracle Blogs, 2 Jan. 2023, blogs.oracle.com/javamagazine/post/java-json-serialization-jackson.

FasterXML. “FasterXML/Jackson: Main Portal Page for the Jackson Project.” GitHub, github.com/FasterXML/jackson. Accessed 28 Feb. 2025.

“Maven Repository: Com.Fasterxml.Jackson.Core » Jackson-Core.” Mvn Repository, mvnrepository.com/artifact/com.fasterxml.jackson.core/jackson-core. Accessed 2 Mar. 2025.

Saloranta, Tatu. Brief History of Jackson the JSON Processor, CowTalk, 8 Aug. 2013, www.cowtowncoder.com/blog/archives/2013/08/entry_479.html.