Alisa Steensen 5/14/25

Module 11.2 Advanced Java Programming

GSON

Gson is a reliable, lightweight choice for developers as a Java JSON API. Gson is a Java library that can be used to convert Java Objects into their JSON representation. (Github n.d.) Additionally, it enables converting a JSON string into a Java object. One of the best advantages of the GSON is the ability to work with arbitrary Java objects, which include those for which you don’t have the source code. Gson was created by Google as an open source, that was first introduced publicly in 2008. The library became a popular choice for a quick and easy way to use JSON parsing library by Java developers. Over the years, it has remained one of the most widely adopted JSON libraries within the Java ecosystem across the globe.

GSON has many key features that make it stand out among other JSON libraries. First, there is no annotations required. While other libraries require Java annotations to enable JSON conversion, GSON does not require modifying classes to enable JSON conversion. This is extremely beneficial if you don’t have access to the original source code. Second, is the support for Java generics. GSON allows for fully supported generic types. This gives flexibility when working with complex data structures. Third, GSON is designed to work seamlessly with existing or unmodified Java objects. Fourth, is that it allows for complex object handling, it can handle deeply nested objects, inheritance hierarchies, and extensive use of generics. Fifth, it allows developers to define custom representations for objects to meet specific requirements. These key features make Gson a stand out API for developers to use.

Some of the Advantages of using GSON include its easy handling within java applications, it’s high level of customization for serialization and deserialization, the unmatched performance is optimized for memory use, and it provides rich annotation support. However, there are some disadvantages that include but are not limited to it being exclusively java based, and cannot be used outside of Java, harder to learn how to use especially for advanced customization, and the external dependency may require updates regularly to maintain compatibility.

Gson is currently in maintenance mode, which means that while existing bugs are being addressed, significant new features are unlikely to be introduced. And while Gson primarily focuses on Java, it may work with other JVM languages like Kotlin or Scala. However, some language specific features, such as Kotlin's non null types or default constructor arguments are not fully supported. Therefore, applications using other JVM languages will need to consider using JSON libraries that fully support those languages to avoid unexpected behaviors.

To use Gson in your projects, you will need to add it as a dependency via Gradle or Maven. Gson Jar files can be downloaded from Maven Central.

Link for Jar Files: <https://repo1.maven.org/maven2/com/google/code/gson/gson/2.10.1/>

Overall, Gson provides an efficient easy to use open-source platform for developers to convert Java objects into JSON. It’s simple and flexible approach offers straightforward methods such as toJson() and fromJson() that will seamlessly convert between Java objects and Json. It remains a reliable choice for developers, especially in projects that need simplicity and compatability with code.

References:

SourceBae. (n.d.). *What is the difference between Gson and JSON?* <https://sourcebae.com/blog/what-is-the-difference-between-gson-and-json/:contentReference[oaicite:3]{index=3>}

Github. (n.d.). *Gson: A Java serialization/deserialization library to convert Java Objects into JSON and back*. GitHub. <https://github.com/google/gson>