UD08_T02: JSON y YAML

1. Introducción a JSON y YAML

1.1. ¿Qué es JSON?

JSON (JavaScript Object Notation) es un formato ligero de intercambio de datos que es fácil de leer y escribir para los humanos, y fácil de parsear y generar para las máquinas. Es un formato basado en texto que utiliza una estructura de pares clave-valor, similar a los objetos en JavaScript.

1.2. ¿Qué es YAML?

YAML (YAML Ain't Markup Language) es un formato de serialización de datos legible por humanos que se utiliza comúnmente para archivos de configuración y en aplicaciones donde los datos deben ser almacenados o transmitidos. YAML es más expresivo que JSON y permite comentarios, lo que lo hace más adecuado para configuraciones complejas.

1.3. Comparación entre JSON y YAML:

Característica	JSON	YAML
Legibilidad	Buena, pero menos expresivo	Muy buena, más expresivo
Comentarios	No soporta comentarios	Soporta comentarios
Estructura	Basado en pares clave-valor	Basado en indentación
Uso común	APIs, intercambio de datos	Configuraciones, DevOps
Complejidad	Más simple	Más flexible y complejo

2. Generación y Carga de JSON y YAML en Java

Para trabajar con `JSON` y `YAML` en Java, utilizaremos las bibliotecas **Jackson**.

Ampliación Porqué `Jackson` y no `Gson` ?

Tanto **Jackson** como **Gson** son bibliotecas populares en Java para trabajar con **JSON** (serialización y deserialización). Ambas son ampliamente utilizadas, pero tienen diferencias en términos de rendimiento, funcionalidades, flexibilidad y facilidad de uso.

- **Jackson**, desarrollada por **FasterXML**, es más potente, flexible y rápido, pero tiene una curva de aprendizaje más pronunciada. Es ideal para proyectos complejos o cuando necesitas trabajar con múltiples formatos (YAML, CSV, etc).
- **Gson**, desarrollada por Google, es más sencilla y fácil de usar, pero menos flexible y potente. Es ideal para proyectos pequeños o cuando necesitas una solución rápida y ligera.

Para trabajar con `JSON` y `YAML` en Java usando **Jackson**, asegúrate de agregar las dependencias correctas en tu `pom.xml`. Jackson es una biblioteca poderosa y flexible que te permite manejar ambos formatos de manera eficiente.

Dependencias Maven:

```xml
    <dependencies>
        <!-- https://mvnrepository.com/artifact/com.fasterxml.jackson.core/jackson-databind -->
        <!-- Dependencia para trabajar con JSON -->
        <dependency>
            <groupId>com.fasterxml.jackson.core</groupId>
            <artifactId>jackson-databind</artifactId>
            <version>2.15.2</version> <!-- Usa la versión más reciente -->
        </dependency>
        <!-- https://mvnrepository.com/artifact/com.fasterxml.jackson.dataformat/jackson-dataformat-yaml -->
        <!-- Dependencia para trabajar con YAML -->
        <dependency>
            <groupId>com.fasterxml.jackson.dataformat</groupId>
            <artifactId>jackson-dataformat-yaml</artifactId>
            <version>2.15.2</version> <!-- Usa la versión más reciente -->
        </dependency>
    </dependencies>
```

Ejemplo Simple: Generación y Carga de JSON y YAML

Clase `Libro`:

```java
public class Libro {
    private String titulo;
    private String autor;
    private int anyoPublicacion;

    //Constructor por defecto, necesario para Jackson
    public Libro() {
    }

    // Constructor, getters y setters
    public Libro(String titulo, String autor, int anyoPublicacion) {
        this.titulo = titulo;
        this.autor = autor;
        this.anyoPublicacion = anyoPublicacion;
    }

    public String getTitulo() {
        return titulo;
    }

    public void setTitulo(String titulo) {
        this.titulo = titulo;
```

```
    }

    public String getAutor() {
        return autor;
    }

    public void setAutor(String autor) {
        this.autor = autor;
    }

    public int getAnyoPublicacion() {
        return anyoPublicacion;
    }

    public void setAnyoPublicacion(int anyoPublicacion) {
        this.anyoPublicacion = anyoPublicacion;
    }

    @Override
    public String toString() {
        return "Libro{" +
                "titulo='" + titulo + '\'' +
                ", autor='" + autor + '\'' +
                ", año Publicacion=" + anyoPublicacion +
                '}';
    }
}
```

```
import com.fasterxml.jackson.databind.ObjectMapper;
import com.fasterxml.jackson.dataformat.yaml.YAMLFactory;

import java.io.File;
import java.io.IOException;
import java.util.ArrayList;

public class TestLibro {
    public static void main(String[] args) {
        // Crear una lista de libros
        ArrayList<Libro> libros = new ArrayList<>();
        libros.add(new Libro("El Principito", "Antoine de Saint-Exupéry", 1943));
        libros.add(new Libro("Cien Años de Soledad", "Gabriel García Márquez", 1967));
        libros.add(new Libro("1984", "George Orwell", 1949));

        // Generar JSON con Jackson
        ObjectMapper jsonMapper = new ObjectMapper(); // ObjectMapper de Jackson: permite leer y escribir JSON
        try {
            jsonMapper.writeValue(new File("libros.json"), libros);
        } catch (IOException ex) {
            System.out.println("Error al generar el archivo JSON.");
        }
        System.out.println("Archivo JSON generado con éxito.");

        // Cargar JSON con Jackson
        ArrayList<Libro> librosCargadosJson = null;
        try {
            librosCargadosJson = jsonMapper.readValue(new File("libros.json"), jsonMapper.getTypeFactory().constructCollectionType(ArrayList.class, Libro.class));
        } catch (IOException ex) {
            System.out.println("Error al cargar el archivo JSON.");
        }
        System.out.println("Libros cargados desde JSON:");
        if (librosCargadosJson != null) {
            for (Libro libro : librosCargadosJson) {
                System.out.println(libro);
            }
        } else {
            System.out.println("No se han cargado libros del JSON.");
        }

        // Generar YAML con Jackson
        ObjectMapper yamlMapper = new ObjectMapper(new YAMLFactory());
```

```java
        try {
            yamlMapper.writeValue(new File("libros.yaml"), libros);
        } catch (IOException ex) {
            System.out.println("Error al generar el archivo YAML.");
        }
        System.out.println("Archivo YAML generado con éxito.");

        // Cargar YAML con Jackson
        ArrayList<Libro> librosCargadosYaml = null;
        try {
            librosCargadosYaml = yamlMapper.readValue(new File("libros.yaml"),
jsonMapper.getTypeFactory().constructCollectionType(ArrayList.class, Libro.class));
        } catch (IOException ex) {
            System.out.println("Error al cargar el archivo YAML.");
        }
        System.out.println("Libros cargados desde YAML:");
        if (librosCargadosYaml != null) {
            for (Libro libro : librosCargadosYaml) {
                System.out.println(libro);
            }
        } else {
            System.out.println("No se han cargado libros del YAML.");
        }
    }
}
```

Resultado Esperado:

Archivo `libros.json`:

```json
[
  {
    "titulo": "El Principito",
    "autor": "Antoine de Saint-Exupéry",
    "anyoPublicacion": 1943
  },
  {
    "titulo": "Cien Años de Soledad",
    "autor": "Gabriel García Márquez",
    "anyoPublicacion": 1967
  },
  {
    "titulo": "1984",
    "autor": "George Orwell",
    "anyoPublicacion": 1949
  }
]
```

Archivo `libros.yaml`:

```yaml
---
- titulo: "El Principito"
  autor: "Antoine de Saint-Exupéry"
  anyoPublicacion: 1943
- titulo: "Cien Años de Soledad"
  autor: "Gabriel García Márquez"
  anyoPublicacion: 1967
- titulo: "1984"
  autor: "George Orwell"
  anyoPublicacion: 1949
```

Salida en Consola:

```
Archivo JSON generado con éxito.
Libros cargados desde JSON:
Libro{titulo='El Principito', autor='Antoine de Saint-Exupéry', año Publicacion=1943}
Libro{titulo='Cien Años de Soledad', autor='Gabriel García Márquez', año Publicacion=1967}
Libro{titulo='1984', autor='George Orwell', año Publicacion=1949}
Archivo YAML generado con éxito.
Libros cargados desde YAML:
Libro{titulo='El Principito', autor='Antoine de Saint-Exupéry', año Publicacion=1943}
Libro{titulo='Cien Años de Soledad', autor='Gabriel García Márquez', año Publicacion=1967}
Libro{titulo='1984', autor='George Orwell', año Publicacion=1949}
```

3. Actividades

Genera un nuevo proyecto, **crea una clase** `Producto` con los siguientes atributos:

- `nombre` (String)
- `precio` (double)
- `stock` (int)

Crear otra clase `TestProducto` **con una lista de objetos** `Producto` con al menos 3 productos.

Genera y guarda la lista en un archivo JSON (`productos.json`) y en un archivo YAML (`productos.yaml`).

Carga y muestra los datos desde ambos archivos.

Genera un zip con el proyecto de IntelliJ. Envía el archivo zip a la tarea de Aules.

4. Fuentes de información

- https://awsacademyinstructure.com
- https://www.deepseek.com/