Rapport du Projet

Rapport du Projet : Liaison de Schéma et Génération de JSON via JAXB dans Spring Boot

# Overvieaw

Ce projet Spring Boot fournit un cadre pour générer dynamiquement un fichier pom.xml qui lie des fichiers XSD (XML Schema Definition) à des classes Java en utilisant JAXB. Il prend également en charge la conversion de ces objets générés par JAXB en JSON et expose les utilitaires associés via des points de terminaison REST.

# Configuration du Serveur

- Port : 8098

- URL de base : http://localhost:8098

# Gestion des fichiers XSD

- Emplacement : Tous les fichiers .xsd sont placés dans src/main/resources/xsd.

- Objectif : Générer des classes Java via JAXB, qui sont ensuite sérialisées en JSON.

**Dépendances**

• spring-boot-starter-web : Pour la création d’API RESTful avec Spring MVC.

• spring-boot-starter-test : Fournit un support de test pour les applications Spring Boot.

• jakarta.xml.bind-api : API JAXB (version Jakarta) pour la liaison XML/Java.

• jaxb-runtime : Runtime nécessaire à JAXB pour le (dé)marshalling XML.(Marshall and Unmarshall

• jaxb-api (version javax) : Ancienne API JAXB pour la compatibilité avec d’anciens outils.

• classgraph : Utilisé pour analyser les packages Java et découvrir les classes.

• reflections : Analyse du classpath et métadonnées des classes.(la dépendance classgraph est meilleurz que reflections)

• jackson-module-jaxb-annotations : Permet à Jackson de comprendre les annotations JAXB.

• jmustache : Moteur de template Mustache pour la génération dynamique du fichier pom.xml.(ce n’est pas une definition genrale)

**Plugins de Construction**

• jaxb2-maven-plugin : Lie les fichiers .xsd aux classes Java à la compilation.

• build-helper-maven-plugin : Enregistre les répertoires sources JAXB générés dans le chemin de build.

• maven-compiler-plugin : Assure la compatibilité avec Java 17.

# Fonctionnalités

**Génération** **de pom.xml**

Méthode Statique :

- Exécuter generate-sources-block.bat

Méthode Dynamique :

- Utilise PomGenerationService et un template XML statique ou appeler /api/pom/generate.

(Recommandé) :

- Appeler /generate/pom

- Utilise PomGeneratorMustacheService avec un template Mustache.

## JAXB Output

Les classes générées sont placées sous src/main/java/tn/bz/generated/... selon le nom du fichier .xsd.

**Génération de JSON**

• SchemaBindingUtils : Lie les objets Java générés par JAXB en JSON.

• SchemaService : Liste les classes racines disponibles générées par JAXB.

• SchemaController : Interface REST pour les opérations liées au schéma.

# REST Endpoints

• /api/classes : Liste les classes racines JAXB disponibles

• /api/generate-json : Génère du JSON à partir de la classe sélectionnée (paramètre : key className Value tn.bz.schema.… selon la liste des classes racines JAXB disponibles)

• /api/pom/generate : Génération dynamique de pom.xml

• /generate/pom : Génération dynamique de pom.xml et compilation Maven (Recommandé)

# Output Directory

Les fichiers JSON sont enregistrés dans src/main/resources/json\_output.

# Additional Notes

• XsdEntry (entité) : Contient les métadonnées de chaque fichier .xsd.

• global.xjb : Fichier de personnalisation JAXB pour éviter les conflits de type.