Serveur de Langage Java et Build Maven

On dirait que vous rencontrez deux problèmes liés dans Visual Studio Code (VSCode) : une plainte concernant la nécessité d'au moins Java 21, et des problèmes avec VSCode ne reconnaissant pas certaines définitions d'objets dans votre code Java, même si votre build Maven compile avec succès. Résolvons cela étape par étape.

Le problème survient parce que les extensions Java de VSCode (spécifiquement le Java Language Server) et votre build Maven ne sont peut-être pas alignés en termes de version du Java Development Kit (JDK) ou de configuration du projet. VSCode signale qu'il nécessite JDK 21 pour fonctionner correctement, et les erreurs de parsing suggèrent qu'il n'interprète pas votre code de la même manière que Maven. Voici comment le résoudre :

Étape 1 : Installer JDK 21

Puisque VSCode demande explicitement Java 21, vous devez vous assurer qu'il est installé sur votre système.

- **Télécharger JDK 21** : Obtenez-le à partir d'une source fiable comme le site web d'Oracle ou Adoptium.
- **L'installer** : Suivez les instructions d'installation et notez le chemin d'installation (par exemple, `C:\Program Files\Java\jdk-21` sous Windows ou `/usr/lib/jvm/java-21-openjdk` sous Linux).
- **Vérifier** : Ouvrez un terminal et exécutez `java -version`. Vous devriez voir quelque chose comme :

  ```
  java version "21.0.1" ...
  ```

Étape 2 : Configurer VSCode pour utiliser JDK 21

Vous devez indiquer aux extensions Java de VSCode d'utiliser JDK 21. Voici comment :

1. **Ouvrir les paramètres de VSCode** :
   - Allez dans `File > Preferences > Settings` (ou appuyez sur `Ctrl + ,`).
2. **Définir le chemin du JDK** :
   - Recherchez `java.jdt.ls.java.home`.
   - Entrez le chemin vers votre installation de JDK 21 (par exemple, `C:\Program Files\Java\jdk-21`).
   - Optionnellement, recherchez `java.home` et définissez-le sur le même chemin pour la cohérence.

3. **Éditer `settings.json` (Alternative)** :
    - Ouvrez la palette de commandes (`Ctrl + Shift + P`), tapez “Preferences: Open Settings (JSON)”, et ajoutez :

```
{
    "java.jdt.ls.java.home": "C:\\Program Files\\Java\\jdk-21",
    "java.home": "C:\\Program Files\\Java\\jdk-21"
}
```

    - Ajustez le chemin en fonction de votre système (utilisez des barres obliques `/` pour Linux/Mac).

Cela garantit que le Java Language Server dans VSCode utilise JDK 21, répondant ainsi à l’exigence minimale.

Étape 3 : Définir le JDK du projet dans VSCode

Pour corriger les problèmes de parsing (par exemple, les définitions d’objets non trouvées), assurez-vous que votre projet dans VSCode utilise également JDK 21 :

- Dans `settings.json`, ajoutez ce qui suit pour spécifier JDK 21 pour votre projet :

```
"java.configuration.runtimes": [
    {
        "name": "JavaSE-21",
        "path": "C:\\Program Files\\Java\\jdk-21",
        "default": true
    }
]
```

- Cela aligne le runtime du projet VSCode avec JDK 21, ce qui devrait l’aider à analyser votre code correctement.

Étape 4 : Vérifier la configuration JDK de Maven

Puisque votre build Maven (`mvn compile`) fonctionne bien, il est probablement en utilisant un JDK compatible. Confirmons et alignons-le avec VSCode :

1. **Vérifier le JDK de Maven** :

- Exécutez `mvn -version` dans votre terminal. Recherchez la ligne “Java version”(par exemple, `Java version: 21.0.1`).
- Si ce n’est pas JDK 21, définissez la variable d’environnement `JAVA_HOME` sur le chemin de votre JDK 21 :
  - **Windows** : `set JAVA_HOME=C:\Program Files\Java\jdk-21`
  - **Linux/Mac** : `export JAVA_HOME=/usr/lib/jvm/java-21-openjdk`
- Exécutez `mvn -version` à nouveau pour confirmer.

2. **Spécifier le JDK dans** `pom.xml` (optionnel) :
   - Si vous souhaitez imposer JDK 21 dans votre projet Maven, ajoutez ceci à votre `pom.xml` :

```
<properties>
    <maven.compiler.source>21</maven.compiler.source>
    <maven.compiler.target>21</maven.compiler.target>
</properties>
```

Étape 5 : Rafraîchir VSCode

Après avoir apporté ces modifications, VSCode a besoin d’un rafraîchissement pour les appliquer :

- Recharger VSCode :
  - Ouvrez la palette de commandes (`Ctrl + Shift + P`) et sélectionnez “Developer: Reload Window” .
- Alternativement, fermez et rouvrez VSCode.

Étape 6 : Assurez-vous que les dépendances sont chargées

Si VSCode ne trouve toujours pas les définitions d’objets, il se peut qu’il n’ait pas chargé correctement les dépendances Maven :

1. **Nettoyer l’espace de travail** :
   - Ouvrez la palette de commandes (`Ctrl + Shift + P`).
   - Exécutez “Java: Clean Java Language Server Workspace”.
2. **Forcer la compilation** :
   - Exécutez “Java: Force Java Compilation”depuis la palette de commandes.
3. **Vérifier les imports** :

- Assurez-vous que les dépendances Maven de votre projet sont importées en cliquant avec le bouton droit sur votre `pom.xml` dans VSCode et en sélectionnant “Update Project Configuration”.

Dépannage

Si le problème persiste : - **Vérifier les journaux** : - Ajoutez `"java.trace.server": "verbose"` à `settings.json`. - Ouvrez le panneau de sortie (`Ctrl + Shift + U`) et sélectionnez “Java Language Server”pour afficher des journaux détaillés. - **Vérifier le code** : - Double-checkez que les définitions d’objets manquantes proviennent de votre code ou des dépendances listées dans `pom.xml`, et non de problèmes externes.

Résumé

En installant JDK 21, en configurant VSCode pour l’utiliser, en alignant vos paramètres de projet et Maven, et en rafraîchissant l’environnement, VSCode devrait cesser de se plaindre de Java 21 et analyser correctement votre code, correspondant à la compilation réussie de Maven. Faites-moi savoir si vous rencontrez des problèmes !