

Maven Surefire Plugin

Voici une explication du **plugin Maven Surefire**, qui est un composant clé dans les projets basés sur Maven pour exécuter des tests unitaires pendant le processus de construction. Cette explication couvre son objectif, sa configuration, son utilisation et ses options de personnalisation.

Qu'est-ce que le plugin Maven Surefire ?

Le **plugin Maven Surefire** est un plugin dans Apache Maven conçu pour exécuter des tests unitaires pendant le cycle de vie de la construction. Il s'intègre de manière transparente avec la phase `test` de Maven et est automatiquement déclenché lorsque vous exécutez des commandes comme `mvn test`, `mvn package`, ou `mvn install`. Le plugin prend en charge les frameworks de test populaires tels que JUnit (versions 3, 4 et 5) et TestNG, et il génère des rapports de tests pour aider les développeurs à évaluer les résultats des tests.

Fonctionnalités Clés

- Exécute les tests dans un processus JVM séparé pour l'isolement.
 - Prend en charge plusieurs frameworks de test (JUnit, TestNG, etc.).
 - Génère des rapports de tests dans des formats comme XML et texte brut.
 - Offre la flexibilité de sauter des tests, d'exécuter des tests spécifiques ou de personnaliser l'exécution.
-

Configuration de Base dans `pom.xml`

Le plugin Surefire est inclus par défaut dans le cycle de vie de construction de Maven, donc vous n'avez pas besoin de le configurer pour une utilisation de base. Cependant, vous pouvez le déclarer explicitement dans votre fichier `pom.xml` pour spécifier une version ou personnaliser son comportement.

Configuration Minimale

Si vous n'ajoutez aucune configuration, Maven utilise le plugin avec les paramètres par défaut : - Les tests sont situés dans `src/test/java`. - Les fichiers de test suivent des motifs de nommage comme `**/*Test.java`, `**/Test*.java`, ou `**/*Tests.java`.

Déclaration Explicite

Pour personnaliser le plugin ou garantir une version spécifique, ajoutez-le à la section `<build><plugins>` de votre `pom.xml`. Voici un exemple :

```
<build>
  <plugins>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-surefire-plugin</artifactId>
      <version>3.0.0-M5</version> <!-- Utilisez la dernière version -->
    </plugin>
  </plugins>
</build>
```

Exécution des Tests avec Surefire

Le plugin est lié à la phase `test` du cycle de vie de Maven. Voici comment l'utiliser :

Exécuter Tous les Tests

Pour exécuter tous les tests unitaires, exécutez :

```
mvn test
```

Exécuter des Tests dans une Construction Plus Large

Les tests sont automatiquement exécutés lorsque vous exécutez des commandes qui incluent la phase `test`, comme :

```
mvn package
```

```
mvn install
```

Ignorer les Tests

Vous pouvez ignorer l'exécution des tests en utilisant des indicateurs de ligne de commande : - **Ignorer l'exécution des tests** : `-DskipTests mvn package -DskipTests` - **Ignorer la compilation et l'exécution des tests** : `-Dmaven.test.skip=true mvn package -Dmaven.test.skip=true`

Personnalisation du Plugin Surefire

Vous pouvez adapter le comportement du plugin en ajoutant une section `<configuration>` dans le `pom.xml`.
Voici quelques personnalisations courantes :

Inclure ou Exclure des Tests Spécifiques

Spécifiez quels tests exécuter ou ignorer en utilisant des motifs :

```
<configuration>
  <includes>
    <include>**/My*Test.java</include>
  </includes>
  <excludes>
    <exclude>**/SlowTest.java</exclude>
  </excludes>
</configuration>
```

Exécuter des Tests en Parallèle

Accélérez l'exécution en exécutant les tests de manière concurrente :

```
<configuration>
  <parallel>methods</parallel>
  <threadCount>2</threadCount>
</configuration>
```

Note : Assurez-vous que vos tests sont thread-safe avant d'activer cette option.

Passer des Propriétés Système

Définissez des propriétés pour la JVM de test :

```
<configuration>
  <systemPropertyVariables>
    <propertyName>propertyValue</propertyName>
  </systemPropertyVariables>
</configuration>
```

Générer des Rapports

Par défaut, les rapports sont enregistrés dans `target/surefire-reports`. Pour un rapport HTML, utilisez le `maven-surefire-report-plugin` :

```
<reporting>
  <plugins>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-surefire-report-plugin</artifactId>
      <version>3.0.0-M5</version>
    </plugin>
  </plugins>
</reporting>
```

Exécutez `mvn surefire-report:report` pour générer le rapport HTML.

Gestion des Échecs de Tests

Échouer la Construction en Cas d'Échec de Test

Par défaut, un test échoué provoque l'échec de la construction. Pour ignorer les échecs et continuer :

```
mvn test -Dmaven.test.failure.ignore=true
```

Réexécuter les Tests Échoués

Gérez les tests capricieux en réessayant les échecs :

```
<configuration>
  <rerunFailingTestsCount>2</rerunFailingTestsCount>
</configuration>
```

Cela réexécute les tests échoués jusqu'à 2 fois.

Utilisation de Surefire avec des Frameworks de Test

Le plugin prend en charge divers frameworks de test avec une configuration minimale :

JUnit 4

Aucune configuration supplémentaire n'est nécessaire ; Surefire détecte automatiquement les tests JUnit 4.

JUnit 5

Ajoutez la dépendance JUnit 5 :

```
<dependencies>
  <dependency>
    <groupId>org.junit.jupiter</groupId>
    <artifactId>junit-jupiter-engine</artifactId>
    <version>5.8.1</version>
    <scope>test</scope>
  </dependency>
</dependencies>
```

Utilisez la version Surefire 2.22.0 ou ultérieure pour une prise en charge complète.

TestNG

Ajoutez la dépendance TestNG :

```
<dependencies>
  <dependency>
    <groupId>org.testng</groupId>
    <artifactId>testng</artifactId>
    <version>7.4.0</version>
    <scope>test</scope>
  </dependency>
</dependencies>
```

Surefire exécutera automatiquement les tests TestNG.

Notes Clés

- **Emplacement des Tests** : Les tests doivent être dans `src/test/java`, avec les ressources dans `src/test/resources`.

- **Conventions de Nommage** : Les motifs par défaut incluent `**/*Test.java`, `**/Test*.java`, etc., mais ceux-ci peuvent être personnalisés.
 - **Version du Plugin** : Utilisez une version récente (par exemple, 3.0.0-M5) pour les dernières fonctionnalités et la prise en charge des frameworks.
-

Résumé

Le **plugin Maven Surefire** est essentiel pour exécuter des tests unitaires dans les projets Maven. Pour l'utiliser : 1. Comptez sur son comportement par défaut ou déclarez-le explicitement dans `pom.xml` pour la personnalisation. 2. Exécutez les tests avec `mvn test` ou dans le cadre de constructions plus grandes comme `mvn package`. 3. Personnalisez-le avec des options comme des motifs d'inclusion/exclusion, une exécution parallèle ou des propriétés système. 4. Gérez les échecs en sautant les tests, en les ignorant ou en réexécutant les tests capricieux. 5. Assurez la compatibilité avec votre framework de test (JUnit, TestNG) en ajoutant les dépendances appropriées.

Ce plugin fournit une méthode robuste et flexible pour gérer les tests unitaires dans vos projets Maven !