Using Liquibase with Spring Boot

Samuel Catalano · Follow

Published in The Fresh Writes

3 min read · Jan 14, 2023

Liquibase is a popular tool for managing and tracking database changes in a Spring Boot project. It allows developers to easily version control their database schema and collaborate with other team members. In this article, we will go over the basics of using Liquibase in a Spring Boot project, specifically by using YAML as the format for the changelog.

1. Adding the Liquibase dependency

The first step in using Liquibase in a Spring Boot project is to add the Liquibase dependency to your project. You can do this by adding the following to your `pom.xml` file:

```xml
<dependency>
    <groupId>org.liquibase</groupId>
    <artifactId>liquibase-core</artifactId>
    <version>3.10.2</version>
</dependency>
```

2. Creating a Changelog

Once you have added the Liquibase dependency to your project, you can create a changelog. A changelog is a file that contains a list of all the changes that you want to make to your database schema. In this case, we will use the YAML format for the changelog. You can create a new changelog by creating a new file in your project's resources folder.

3. Configuring Liquibase

After creating the changelog, you need to configure Liquibase to use it. You can do this by adding the following to your `application.yml` file:

```yaml
spring:
  liquibase:
    enabled: true
    change-log: classpath:changelog.yaml
```

This will tell Spring Boot to enable Liquibase and use the `changelog.yaml` file as the changelog.

4. Defining Changes

Now you can define the changes you want to make to your database. For example, you can create a table by adding the following to your changelog:

```yaml
databaseChangeLog:
 - changeSet:
     id: create-table-users
     author: YourName
     changes:
      - createTable:
          tableName: users
          columns:
            - column:
                name: id
                type: INT
                constraints:
                  primaryKey: true
                  nullable: false
            - column:
                name: name
                type: VARCHAR(255)
```

5. Updating your Database

Once you have defined your changes in the changelog, you can update your database by running the Spring Boot application. Liquibase will automatically apply the changes defined in the changelog to the database.

6. Rolling Back Changes

If you ever need to roll back changes that you have made to your database, you can use Liquibase to do so. To roll back a specific change, you can use the `rollback` command in the command line. For example, to roll back the change set, you can run the following command:

```
liquibase rollback create-table-users
```

In conclusion, using Liquibase in a Spring Boot project with YAML is a powerful way to manage and track database changes. It makes it easy to version control your database schema and collaborate with other team members. With the basics covered in this article, you should now be able to start using Liquibase in your Spring Boot project with ease. It is important to note that Liquibase supports multiple formats for changelogs, such as XML, JSON and SQL. Using YAML as the format for the changelog allows for a more human-readable format, making it easier to understand and maintain. Additionally, Spring Boot provides a convenient integration with Liquibase, allowing for automatic updates to the database when the application starts, which eliminates the need for manual updates.

It is also important to keep in mind that Liquibase provides a wide range of features and options, such as the ability to roll back specific changes, generate database documentation, and integrate with popular version control systems such as Git. It is a powerful tool that can help you manage and track database changes in a professional and efficient way.

In summary, Liquibase is a powerful tool that can help you manage and track database changes in a Spring Boot project. By using YAML as the format for the changelog, you can have a more human-readable format, and Spring Boot integration allows for automatic updates to the database when the application starts. With the basics covered in this article, you should now be able to start using Liquibase to manage your database schema with ease.

Thanks for reading.Happy learning 😄

Do support our publication by following it

The Fresh Writes

We support small publishers to enhance their articles and increase their growth

medium.com

Java Concurrent API Packages — Part 1

Concurrent utils are included in the package java.util.concurrent package. This package has two important sub packages…

medium.com

Software I use to plan my writing content