1. Flyway Maven Plugin

Ref: <https://flywaydb.org/getstarted/firststeps/maven>

This method integrates flyway by adding the following into the pom file

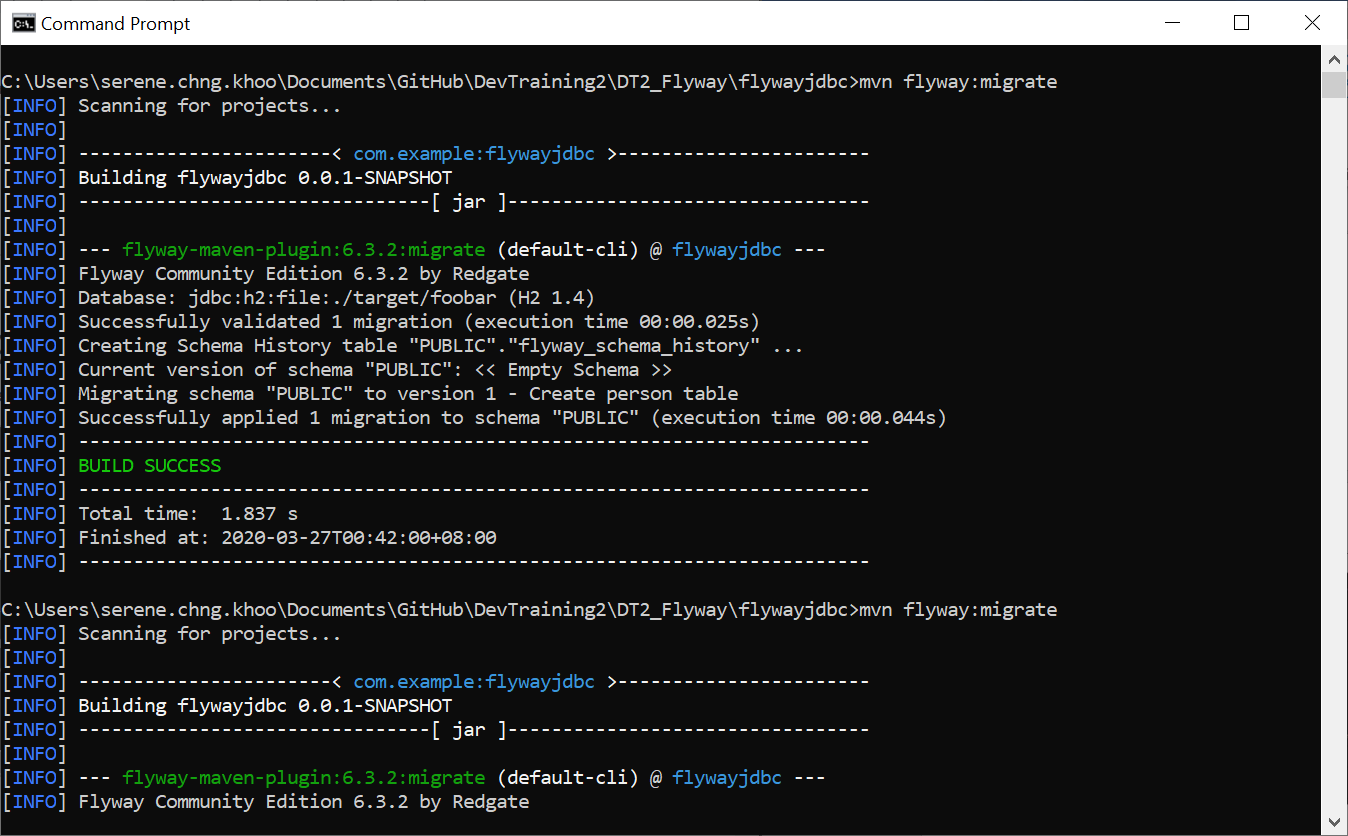
<plugins>

<plugin>  
 <groupId>org.flywaydb</groupId>  
 <artifactId>flyway-maven-plugin</artifactId>  
 <version>6.3.2</version>  
 <configuration>  
 <url>jdbc:h2:file:./target/foobar</url>  
 <user>sa</user>  
 </configuration>  
 <dependencies>  
 <dependency>  
 <groupId>com.h2database</groupId>  
 <artifactId>h2</artifactId>  
 <version>1.4.197</version>  
 </dependency>  
 </dependencies>  
 </plugin>  
</plugins>

Has a migration database in src/main/resources/db/migration/V1\_\_Create\_person\_table.sql

create table PERSON (  
 ID *int* not null,  
 NAME *varchar*(100) not null,  
 AGE *int* null  
);

Is executed as follows



1. Flyway command line tool

Ref: <https://flywaydb.org/getstarted/firststeps/commandline>

This method requires downloading the Flyway Command-line Tool and then editing the config file in the downloaded file - flyway-6.3.2, in this directory /conf/flyway.conf as follows:

flyway.url=jdbc:h2:file:./foobardb

flyway.user=SA

flyway.password=

\* It is not linked to any maven projects whatsoever.

All migration files are placed into the directory of /sql

Eg: V1\_\_Create\_person\_table.sql

create table PERSON (

ID int not null,

NAME varchar(100) not null

);

Executed by cmd \*must be in the flyway-6.3.2 directory first.

Command:

../ flyway-6.3.2> flyway migrate

Successful output:

Database: jdbc:h2:file:./foobardb (H2 1.4)

Successfully validated 1 migration (execution time 00:00.008s)

Creating Schema History table: "PUBLIC"."flyway\_schema\_history"

Current version of schema "PUBLIC": << Empty Schema >>

Migrating schema "PUBLIC" to version 1 - Create person table

Successfully applied 1 migration to schema "PUBLIC" (execution time 00:00.033s)

1. Integrating spring boot with flyway

Ref: <https://dzone.com/articles/database-versioning-with-flyway-and-java>

This method requires to add the following to the pom file

<dependency>  
 <groupId>org.flywaydb</groupId>  
 <artifactId>flyway-core</artifactId>  
</dependency>  
<dependency>  
 <groupId>org.hsqldb</groupId>  
 <artifactId>hsqldb</artifactId>  
 <version>2.3.3</version>  
</dependency>

And the following to the application.properties file

# defining location for HSQLDB's data  
spring.datasource.url=jdbc:hsqldb:file:data/app  
# disabling Hibernate's auto schema generation  
spring.jpa.hibernate.ddl-auto=none

I’m not sure how exactly flyway is being executed here, but the tutorial is about a RESTful application that uses flyway to create the database and update its database

So it has the typical migration script in the src/main/resources/db/migration/ folder, ie:

**V1\_\_customers.sql**

create table customer (

id int identity primary key,

name varchar(255) not null,

contact\_name varchar (255) not null,

email varchar (255) not null,

phone varchar (255) not null

);

insert into customer (name, contact\_name, email, phone) values

('Coca Cola', 'John Doe', 'john.doe@cocacola.com', '202-555-0143'),

('Dell', 'Bob Frapples', 'bob.frapples@dell.com', '202-555-0180'),

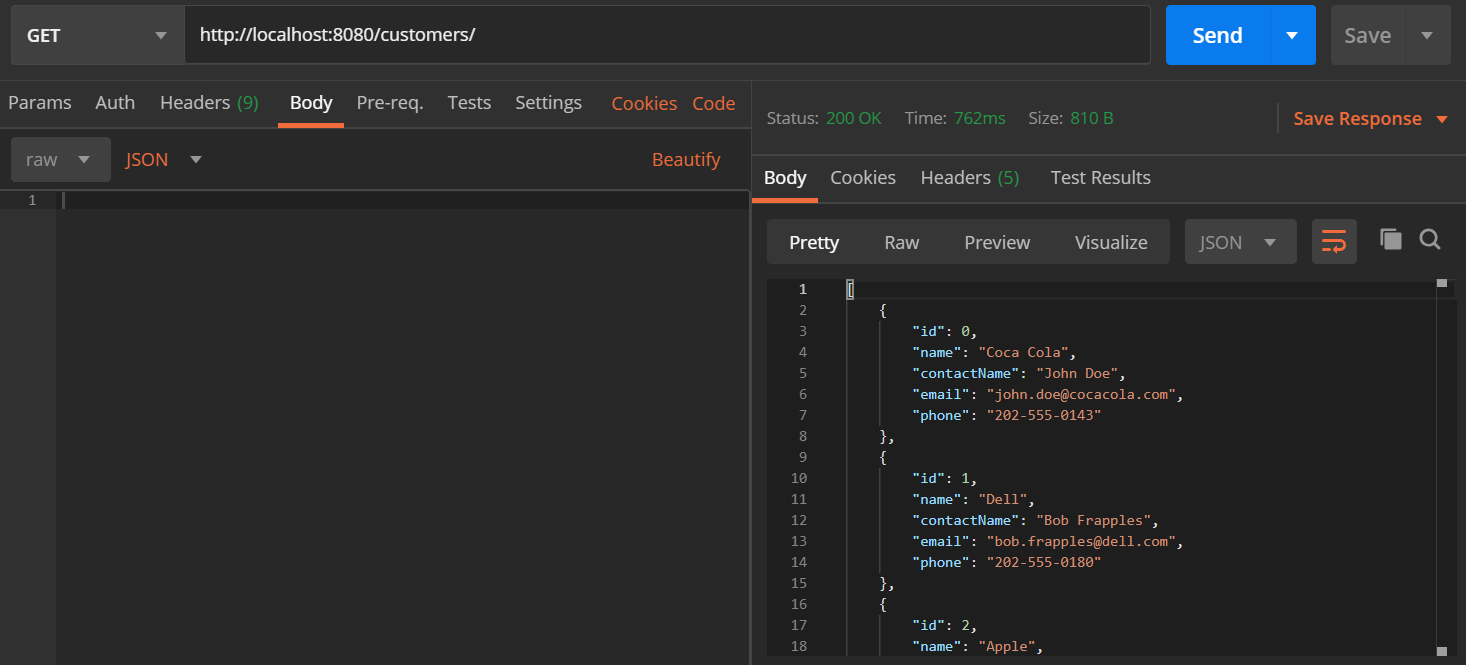
('Apple', 'Barb Ackue', 'barb.ackue@apple.com', '202-555-0128'),

('Google', 'Sue Vaneer', 'sue.vaneer@google.com', '202-555-0174'),

('FedEx', 'Robin Banks', 'robin.banks@fedex.com', '202-555-0146'),

('Salesforce', 'Zack Lee', 'zack.lee@salesforce.com', '202-555-0122');

So in postman, you can see the results as follow



Conclusion:

After looking at all 3 methods, I think the one we’re meant to do is the first one, but, feel free to correct me if I’m wrong, and if you don’t mind, you can share with me how you guys did your flyway.

Question:

How do you guys find the flyway\_schema\_history? I can’t seem to google the answer lol

Answers:

[3:37 PM] Theng, Wai Loon

Ch'ng Khoo, Serene: Good! Nice to see thorough findings. Do #1 and #3, each for different purposes.

1. #3 is to allow your Spring Boot to execute Flyway migration everytime when your Spring Boot app starts running. This allows configuration of Flyway settings using application.properties.
2. #1 functionality is slightly overlapping with #3, but it gives the ability to execute flyway migration using command line if required, e.g. mvn flyway:migrate. The command line use case might not be for developers; instead it's often used in CICD pipelines.