(/)

Spring Boot With H2 Database

Last modified: September 13, 2020

by baeldung (https://www.baeldung.com/author/baeldung/)

Persistence (https://www.baeldung.com/category/persistence/)
Spring Boot (https://www.baeldung.com/category/spring/spring-boot/)

1. Overview

In this tutorial, we'll explore using H2 with Spring Boot. Just like other databases, there's full intrinsic support for it in the Spring Boot ecosystem.

Further reading:

List of In-Memory Databases (/java-in-memory-databases)

A quick review of how to configure some of the more popular in-memory databases for a Java application.

Read more (/java-in-memory-databases) →

Spring Boot with Hibernate (/spring-boot-hibernate)

A quick, practical intro to integrating Spring Boot and Hibernate/JPA.

Read more (/spring-boot-hibernate) →

2. Dependencies

Let's begin with the *h2 (https://search.maven.org/search?q=g:com.h2database%20a:h2)* and *spring-boot-starter-data-jpa (https://search.maven.org/search?q=a:spring-boot-starter-data-jpa%20g:org.springframework.boot)* dependencies:

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>
<dependency>
    <groupId>com.h2database</groupId>
    <artifactId>h2</artifactId>
    <scope>runtime</scope>
</dependency>
```

3. Database Configuration

By default, Spring Boot configures the application to connect to an in-memory store with the username *sa* and an empty password.

However, we can change those parameters by adding the following properties to the *application.properties* file:

```
spring.datasource.url=jdbc:h2:mem:testdb
spring.datasource.driverClassName=org.h2.Driver
spring.datasource.username=sa
spring.datasource.password=password
spring.jpa.database-platform=org.hibernate.dialect.H2Dialect
```

By design, the in-memory database is volatile, and data will be lost when we restart the application.

We can change that behavior by using file-based storage. To do this we need to update the *spring.datasource.url*:

```
spring.datasource.url=jdbc:h2:file:/data/demo
```

The database can also operate in other modes (http://www.h2database.com/html/features.html#connection_modes).

4. Database Operations

Carrying out CRUD operations with H2 within Spring Boot is the same as with other SQL databases, and our tutorials in the Spring Persistence (/persistence-with-spring-series) series do a good job of covering this.

In the meantime, let's add a *data.sql* file in *src/main/resources*:

```
DROP TABLE IF EXISTS billionaires;

CREATE TABLE billionaires (
  id INT AUTO_INCREMENT  PRIMARY KEY,
  first_name VARCHAR(250) NOT NULL,
  last_name VARCHAR(250) NOT NULL,
  career VARCHAR(250) DEFAULT NULL
);

INSERT INTO billionaires (first_name, last_name, career) VALUES
  ('Aliko', 'Dangote', 'Billionaire Industrialist'),
  ('Bill', 'Gates', 'Billionaire Tech Entrepreneur'),
  ('Folrunsho', 'Alakija', 'Billionaire Oil Magnate');
```

Spring Boot will automatically pick up the *data.sql* and run it against our configured H2 database during application startup. This is a good way to seed the database for testing or other purposes.

5. Accessing the H2 Console

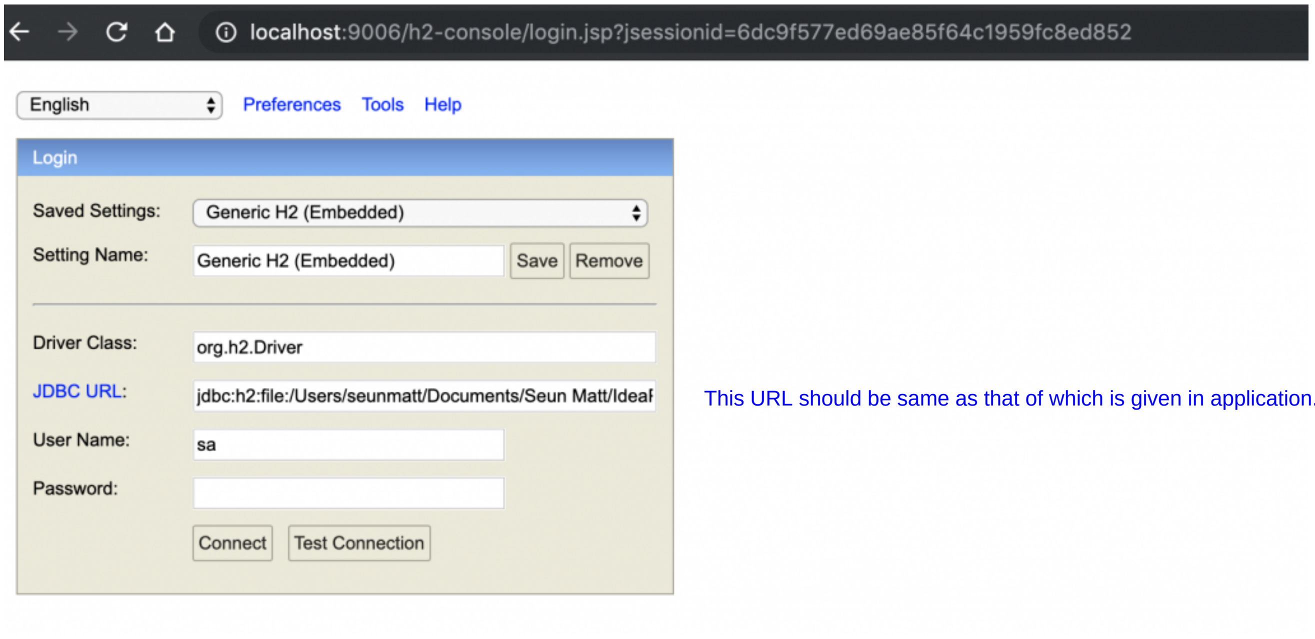
H2 database has an embedded GUI console for browsing the contents of a database and running SQL queries. By default, the H2 console is not enabled in Spring.

To enable it, we need to add the following property to *application.properties*:

```
spring.h2.console.enabled=true
```

Then, after starting the application, we can navigate to *http://localhost:8080/h2-console*, which will present us with a login page.

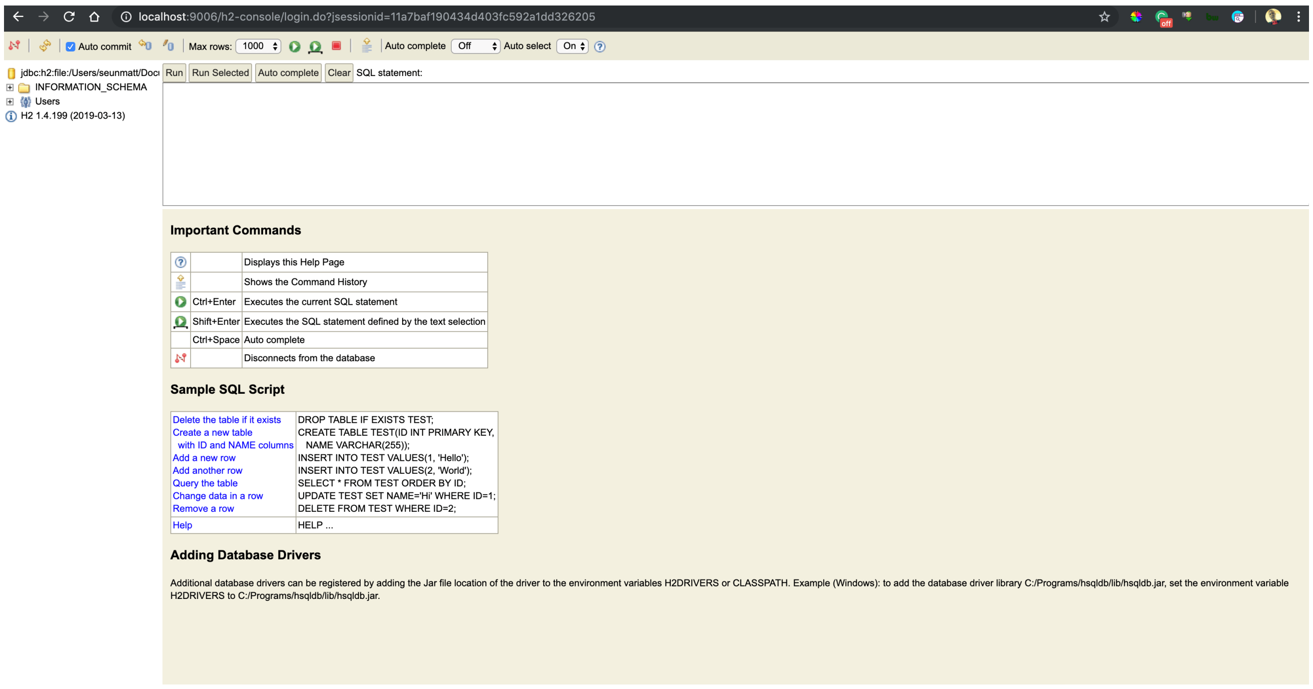
On the login page, we'll supply the same credentials that we used in the *application.properties*:


localhost:9006/h2-console/login.jsp?jsessionid=6dc9f577ed69ae85f64c1959fc8ed852
English
Preferences
Tools
Help
Login
Save
Remove
Connect
Test Connection

Saved Settings: Generic H2 (Embedded)
Setting Name: Generic H2 (Embedded)
Driver Class: org.h2.Driver
JDBC URL: jdbc:h2:file:/Users/seunmatt/Documents/Seun Matt/Ideaf
User Name: sa
Password:

This URL should be same as that of which is given in application.properties file

(/wp-content/uploads/2019/04/Screenshot-2019-04-13-at-5.21.34-PM-e1555173105246-1024x496.png)
Once we connect, we'll see a comprehensive webpage that lists all the tables on the left side of the page and a textbox for running SQL queries:



localhost:9006/h2-console/login.do?jsessionid=11a7baf190434d403fc592a1dd326205

Auto commit ✓ | Max rows: 1000 | Auto complete: Off | Auto select: On

jdbc:h2:file:/Users/seunmatt/Doc
INFORMATION_SCHEMA
Users
H2 1.4.199 (2019-03-13)

Run | Run Selected | Auto complete | Clear | SQL statement:

Important Commands

?		Displays this Help Page
		Shows the Command History
	Ctrl+Enter	Executes the current SQL statement
	Shift+Enter	Executes the SQL statement defined by the text selection
	Ctrl+Space	Auto complete
		Disconnects from the database

Sample SQL Script

Delete the table if it exists	DROP TABLE IF EXISTS TEST;
Create a new table with ID and NAME columns	CREATE TABLE TEST(ID INT PRIMARY KEY, NAME VARCHAR(255));
Add a new row	INSERT INTO TEST VALUES(1, 'Hello');
Add another row	INSERT INTO TEST VALUES(2, 'World');
Query the table	SELECT * FROM TEST ORDER BY ID;
Change data in a row	UPDATE TEST SET NAME='Hi' WHERE ID=1;
Remove a row	DELETE FROM TEST WHERE ID=2;
Help	HELP ...

Adding Database Drivers

Additional database drivers can be registered by adding the Jar file location of the driver to the environment variables H2DRIVERS or CLASSPATH. Example (Windows): to add the database driver library C:/Programs/hsqldb/lib/hsqldb.jar, set the environment variable H2DRIVERS to C:/Programs/hsqldb/lib/hsqldb.jar.

(/wp-content/uploads/2019/04/Screenshot-2019-04-13-at-5.25.16-PM.png)
The web console has an auto-complete feature that suggests SQL keywords. The fact that the console is lightweight makes it handy for visually inspecting the database or executing raw SQL directly.

Moreover, we can further configure the console by specifying the following properties in the project's *application.properties* with our desired values:

```
spring.h2.console.path=/h2-console
spring.h2.console.settings.trace=false
spring.h2.console.settings.web-allow-others=false
```

In the snippet above, we set the console path to be */h2-console*, which is relative to the address and port of our running application. Therefore, if our app is running at *http://localhost:9001*, the console will be available at *http://localhost:9001/h2-console*.

Furthermore, we set *spring.h2.console.settings.trace* to *false* to prevent trace output, and we can also disable remote access by setting *spring.h2.console.settings.web-allow-others* to *false*.

6. Conclusion