**Let’s try to connect to H2 Database**

We can define some properties or configurations inside our application.

For the same, we need to go to these resource folders and this resource folder.

We have application.properties file.

This file inside this property file only we can define all the configurations related to Spring Boot where we can mention the port number which needs to be considered by the spring boot, to start the service, we can define database related configurations, we can define message queues, related configurations, we can define actuator related configurations.

So there are good amount of properties or configurations that we can define inside this file, which will make developer life easy.

With the properties that we mentioned inside this file, Spring Boot is going to do a lot of work behind the scenes. So instead of we writing the Java code or lot of boilerplate code to handle all those logic, we simply mention the properties inside this file and Spring Boot is going to take care of connecting to database, starting the server at the mentioned port number and enabling certain features of actuator.

This way we can achieve many functionalities with these properties, but we are not going to write our properties with the format of key and value, which is the default way inside spring boot.

Instead, we are going to define all our properties with the Yaml format. Let’s rename the application.properties to application.yml yml indicates that this file is going to support the yml format.

let's try to define the properties with the help of .yml. But here you may have questions like how can I define properties with the help of Yaml format?

What is the syntax that I need to follow?

If you see the below properties file this file, these are the normal property values that I can mention in properties format.

spring:.datasource.url=jdbch2:mem:testdb  
spring:.datasource.driver-class-name=org.h2.Driver  
spring:.datasource.username=sa  
spring:.datasource.password=  
spring:.jpa.datasource-platform=org.hibernate.dialect.H2Dialect  
spring:.h2.console.enabled=true  
spring:.jpa.hibernate.ddl-auto=update  
spring:.jpa.show-sql=true

server:  
 port: 8080  
spring:  
 datasource:  
 url: jdbc:h2:mem:testdb  
 username: sa  
 password: ''  
 h2:  
 console:  
 enabled: true  
 jpa:  
 database-platform: org.hibernate.dialect.H2Dialect  
 hibernate:  
 ddl-auto: update  
 show-sql: true

So you can see this is the key and this is the value. So there will be a key and there will be a value.

The same applies for all other properties. Every property will have a key and it is going to have a corresponding value.

So these are the properties and it is very clear for everyone there is no big complexity here there is a key, there is a value.

So now let's try to understand how to represent this inside a yaml,

yaml is going to work based upon indentation.

When I say indentation you can see whenever I want to define this server.port as a key inside yaml

I need to first mention server followed by : post that I should immediately go to a tab space inside

the next row.

Like you can see here I have given a tab space. Then only my Yaml is going to consider the key as server.port.

And very similarly for spring.datasource.url. I need to mention spring followed by a tab space datasource: and since inside data source we are going to have the URL we need to mention the URL again with the tab space. So you can see I have mentioned the tab space in the file above.

The advantage of Yaml, you can see this spring is repeated unnecessarily inside properties file for every key, whereas with yaml it is going to have only once inside the same spring

we can define data source only once, whereas in property you can see it is repeated four times and

at last we reached to the end of the key which is URL. Once we reach, we need to give the : and post the :

we need to make sure we are giving a single space followed by what is the value? This is the value that,

i want to mention the same I have done for driverClassName and user name and password.

And since I want to give empty password, I have given a single quote with empty value. And very similarly, if you see for this property which is spring.h2.console.enabled = true.

I don't need to mention the spring again. I can straight away come to the same location where that data source is present and in the same location. I can start the new key, which is h2.

This way I need to maintain this indentation properly. Then only my Yaml is going to work.

If I screw up this indentation, then my Yaml is not going to work.

Now, like I said, during startup we want some tables to be created so that we can store that data inside these tables.

For the same Under these resources folder, you need to create a new file with the name schema.sql

Whenever you want to create the tables columns, you need to define a file with the name

shema.sql.

Whereas if you want to create data with the help of insert scripts or update scripts, you can define

a file with the name data.sql.

For now, I just want to create the tables inside my internal database.

Below the content in the schema.sql file

CREATE TABLE IF NOT EXISTS `customer` (  
 `customer\_id` int AUTO\_INCREMENT PRIMARY KEY,  
 `name` varchar(100) NOT NULL,  
 `email` varchar(100) NOT NULL,  
 `mobile\_number` varchar(20) NOT NULL,  
 `created\_at` date NOT NULL,  
 `created\_by` varchar(20) NOT NULL,  
 `updated\_at` date DEFAULT NULL,  
 `updated\_by` varchar(20) DEFAULT NULL  
);  
  
CREATE TABLE IF NOT EXISTS `accounts` (  
 `customer\_id` int NOT NULL,  
 `account\_number` int AUTO\_INCREMENT PRIMARY KEY,  
 `account\_type` varchar(100) NOT NULL,  
 `branch\_address` varchar(200) NOT NULL,  
 `created\_at` date NOT NULL,  
 `created\_by` varchar(20) NOT NULL,  
 `updated\_at` date DEFAULT NULL,  
 `updated\_by` varchar(20) DEFAULT NULL  
);

Run your sprint boot application.

So let me try to access this h2 console in the browser.

So I need to go to the localhost 8080 followed by h2 /console and here I'm trying to access it.

You can see you will get this page where you need to enter all your driver class, Jdbc URL, username

and password.

By default, these values will get automatically populated. Please make sure the values are matching with what you have mentioned inside the applications.yml file.

Below the H2 console

<http://localhost:8080/h2-console>



