Getting started JPA with SpringBoot

Project creation

Spring boot provides a web application to create easily a project: http://start.spring.io/

At the beginning choose the default options creating a Gradle project.

Build the project using the command: gradle build

Convert the project into an Eclipse project: gradle eclipse

Then import the project into Eclipse.

Adding JPA libraries to the project

Add the following lines to the file build.gradle:

```
compile group: 'org.eclipse.persistence', name: 'javax.persistence', version: '2.0.0'
compile group: 'hsqldb', name: 'hsqldb', version: '1.8.0.10'
compile group: 'org.eclipse.persistence', name: 'eclipselink', version: '2.5.1'
```

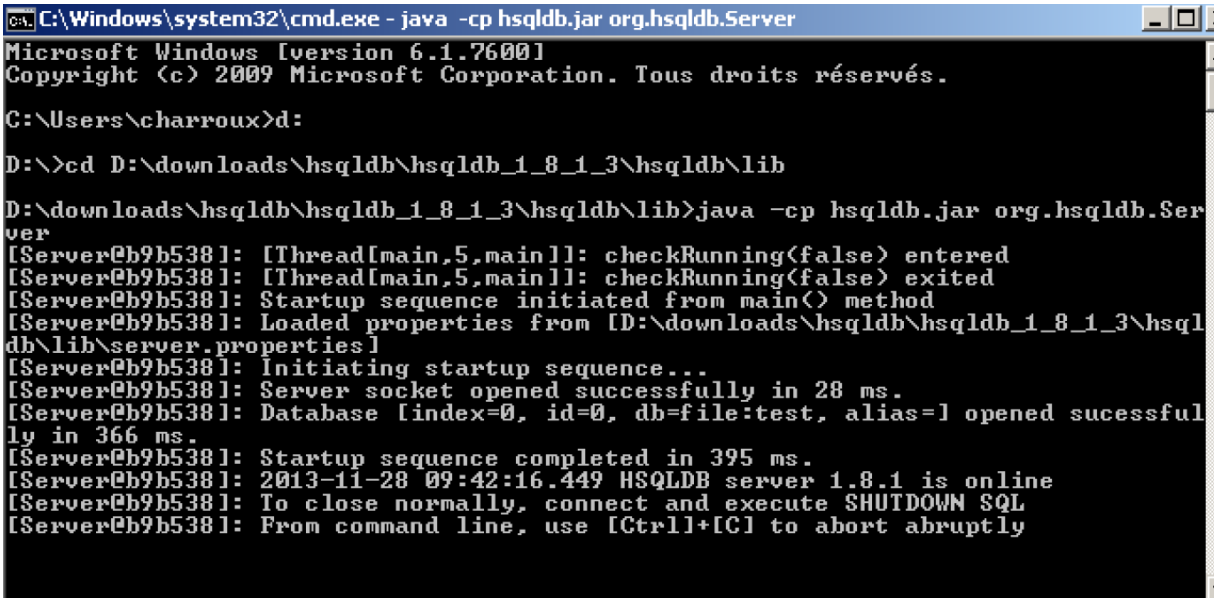
Then rebuilt the project with: “gradle build” and “gradle eclipse”. Don’t forget to refresh the Eclipse project.

Installing and start HSQLDB

The database used is HSQLDB.

From the HSQLDB web site download and unzip HSQLDB version 1.8.

From the lib directory of HSQLDB start the server with:



From the lib directory of HSQLDB start the database manager:

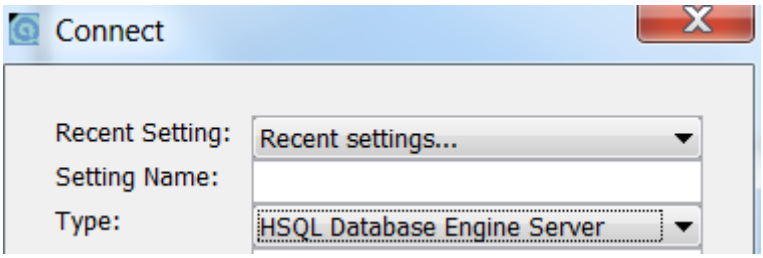
```
Microsoft Windows [version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. Tous droits réservés.

C:\Users\charroux>d:

D:\>cd D:\downloads\hsqldb\hsqldb_1_8_1_3\hsqldb\lib

D:\downloads\hsqldb\hsqldb_1_8_1_3\hsqldb\lib>java -cp hsqldb.jar org.hsqldb.uti
l.DatabaseManagerSwing
Failed to load preferences.  Proceeding with defaults:
```

Choose server in the connection panel.



Structure your code

As usual for Spring Boot, the application should be structured as follow:

```
com
 +- example
     +- myproject
         +- Application.java
         |
         +- domain
         |   +- Customer.java
         |   +- CustomerRepository.java
         |
         +- service
         |   +- CustomerService.java
         |
         +- web
             +- CustomerController.java
```

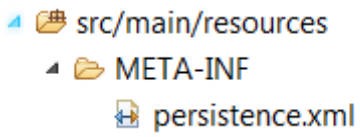
Here, no web layer is needed nether the CustomerRepository. Put the Java persistent classes into the domain package (you can choose the class you want). Look at an example of such a class:

https://github.com/charroux/JPABasis-master/tree/master/src/main/java/com/univ

The service layer will hide JPA with methods. In order to use JPA, put the following lines into your service class:

```
EntityManagerFactory emf = Persistence.createEntityManagerFactory("manager1");
EntityManager entityManager = emf.createEntityManager();
```

manager1 is a persistentUnit, i.e. a database configuration. This configuration is done in a file named persistent.xml contained into the directory resources:



Here is an example of such a configuration:

https://github.com/charroux/JPABasis/tree/master/src/main/resources/META-INF

Write a main program

Spring boot allows adding code into a main program using CommandLineRunner. Here is an example:

```
@SpringBootApplication
public class SampleSimpleApplication implements CommandLineRunner {
	@Bean
	HelloWorldService helloWorldService(){
		return new new HelloWorldServiceImpl();
	}
	@Autowire
	private HelloWorldService helloWorldService;
	@Override
	public void run(String... args) {
		helloWorldService.hello();
	}
}
```

Add to the main program instructions:

- To store objects into the database
- To retrieve object from the database