Spring Roo – A Quick App Development

# Introduction

Spring Roo is a framework of Spring framework that enables quick app development.

There are 2 modes the application may be developed

* Auto generate from database structure
* Generate from spring roo cli

# Requirements

JDK 1.7 minimum

Oracle database

Maven

Following sections perform the steps to develop the application using Oracle as the database

# Auto Generate from database structure

This needs older version of spring roo – v 1.3

1. Download the spring roo v1.3 and add the roo.sh / roo.bat to the path

<http://spring-roo-repository.springsource.org/>

1. Determine the version of the Oracle database to determine the version of the ojdbc driver you need.
2. Oracle integration is not out of the box. It needs the osgi ojdbc jar to be installed. Following links have helped to use roo to osgify the ojdbc jar. Add this osgi jar to the bundle.

<http://rns-techtock.blogspot.com/2012/08/oracle-database-reverse-engineering.html>

<https://kerrywilson.dev/post/2012/01-29-spring-roo-reverse-engineer/>

<https://nidget.wordpress.com/2011/07/21/how-to-osgify-an-oracle-jdbc-driver-with-spring-roo/>

1. Create a new directory for your project and cd into it
2. Go to the spring roo cli using roo.sh / roo.bat
3. Create a project in roo

roo> project --topLevelPackage com.jin.rooapp13

Created ROOT/pom.xml

Created SRC\_MAIN\_RESOURCES

Created SRC\_MAIN\_RESOURCES/log4j.properties

Created SPRING\_CONFIG\_ROOT

Created SPRING\_CONFIG\_ROOT/applicationContext.xml

1. Ensure that the ojdbc bundle is active and available

roo> osgi ps

START LEVEL 99

ID|State |Level|Name

0|Active | 0|System Bundle (4.4.1)

1|Active | 1|file:/Users/deeptivijay/Documents/\_MyLearning\_/springroo/spring-roo-1.3.2.RELEASE/bundle/ST4-4.0.8.jar (0.0.0)

2|Active | 1|file:/Users/deeptivijay/Documents/\_MyLearning\_/springroo/spring-roo-1.3.2.RELEASE/bundle/animal-sniffer-annotations-1.9.jar (0.0.0)

3|Active | 1|antlr-java-parser (1.0.15)

4|Active | 1|file:/Users/deeptivijay/Documents/\_MyLearning\_/springroo/spring-roo-1.3.2.RELEASE/bundle/antlr-runtime-3.5.2.jar (0.0.0)

…

85|Active | 1|Spring User Agent Analysis - Client (1.0.2.RELEASE)

116|Active | 1|com-oracle-convert-osgi (12.1.0.2\_0001)

roo>

1. Create the JPA layer for Oracle using hibernate

roo> jpa setup --database ORACLE --provider HIBERNATE

Created SPRING\_CONFIG\_ROOT/database.properties

The ORACLE JDBC driver is not available in public Maven repositories. Please adjust the pom.xml dependency to suit your needs

Updated SPRING\_CONFIG\_ROOT/applicationContext.xml

Created SRC\_MAIN\_RESOURCES/META-INF/persistence.xml

Updated ROOT/pom.xml [added dependencies com.oracle:ojdbc14:10.2.0.5, org.hibernate:hibernate-core:4.3.6.Final, org.hibernate:hibernate-entitymanager:4.3.6.Final, org.hibernate.javax.persistence:hibernate-jpa-2.1-api:1.0.0.Final, commons-collections:commons-collections:3.2.1, org.hibernate:hibernate-validator:4.3.2.Final, javax.validation:validation-api:1.0.0.GA, javax.transaction:jta:1.1, org.springframework:spring-jdbc:${spring.version}, org.springframework:spring-orm:${spring.version}, commons-pool:commons-pool:1.5.6, commons-dbcp:commons-dbcp:1.4]

roo>

1. Edit the pom.xml for the correct ojdbc jar
2. Edit the database.properties with correct database credentials
3. Run the following command to verify the xml generated by connecting to the database

roo> database introspect --schema foodie

1. Run the following command to generate the database

roo> database reverse engineer --schema foodie

1. Configure the application with a web presentation layer

roo> web mvc setup

1. Generate the controllers for all the entities created

roo> web mvc all --package ~.web

The previous command implements the controllers and view artifacts for all the data access operations (i.e. create, show, list, update and delete), and for all the entities in the persistence layer.

1. Run the application

roo> perform command --mavenCommand tomcat7:run

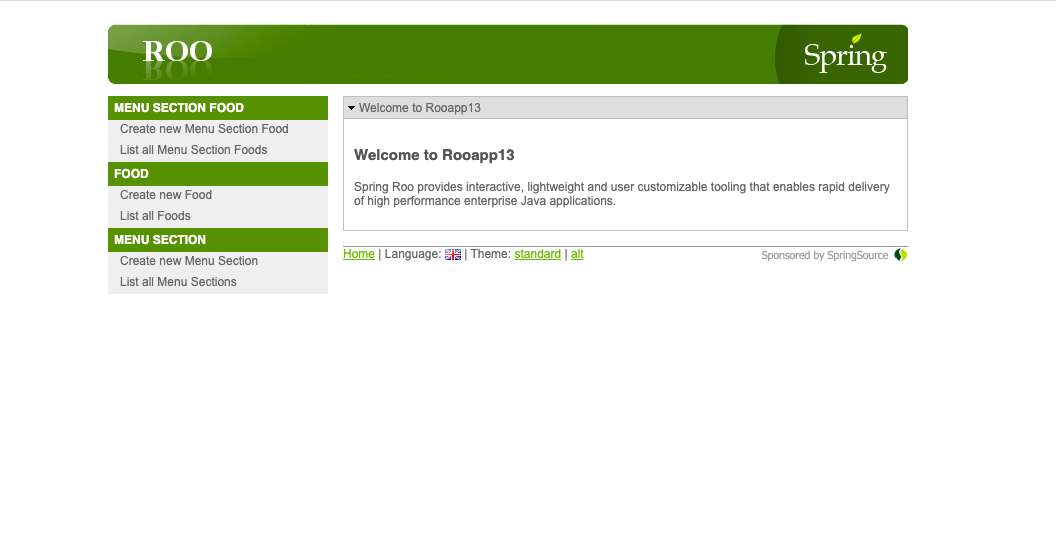
OR

$ mvn tomcat7:run

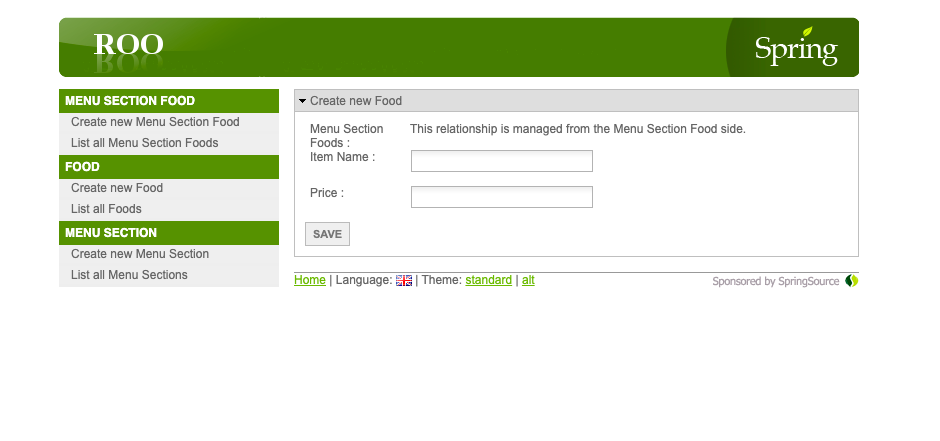
# Application view

<http://localhost:8080/rooapp13/>

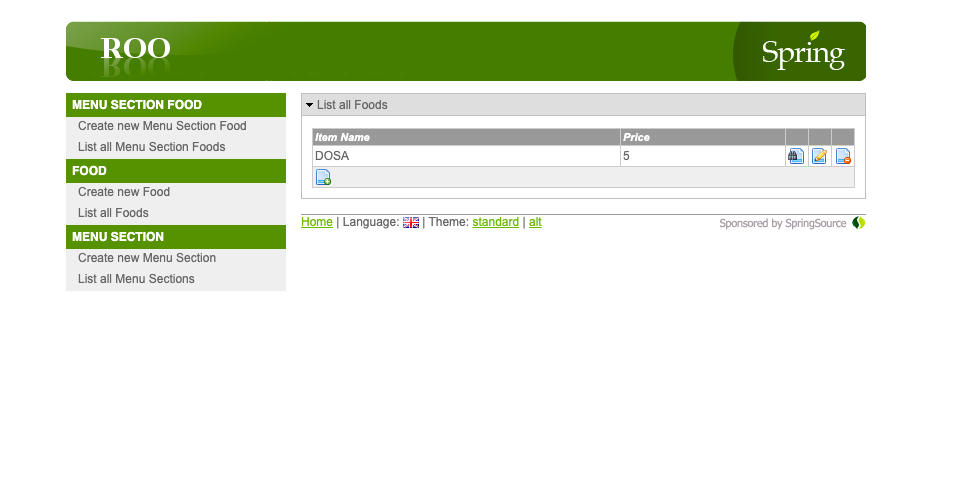
## Homepage



## Create Food



## List Food



# Editing Files

* Do not edit the files marked as generated
* You may edit the java files

After editing, open the roo cli and hold for a moment until roo recognizes the changes done

$ roo.sh

JAVA\_HOME /Library/Java/JavaVirtualMachines/jdk1.8.0\_261.jdk/Contents/Home

\_\_\_\_ \_\_\_\_ \_\_\_\_

/ \_\_ \/ \_\_ \/ \_\_ \

/ /\_/ / / / / / / /

/ \_, \_/ /\_/ / /\_/ /

/\_/ |\_|\\_\_\_\_/\\_\_\_\_/ 1.3.2.RELEASE [rev 8387857]

Welcome to Spring Roo. For assistance press TAB or type "hint" then hit ENTER.

Updated SRC\_MAIN\_JAVA/com/jin/rooapp13/Food\_Roo\_Jpa\_Entity.aj

roo> exit

# Special Scenarios / Features

## Sequence Missing

If the application table has sequences on the primary key and they are not recognized in the dre, define them via the cli or add them to the domain class

@RooJavaBean

@RooJpaActiveRecord(table = "FOOD", schema = "FOODIE", sequenceName = "FOOD\_SEQ")

@RooDbManaged(automaticallyDelete = true)

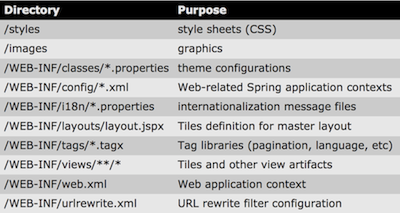
@RooToString(excludeFields = { "menuSectionFoods" })

public class Food {

}

## UI Update

Update the files in the webapp folder like the css, images and jspx



## Authentication

Enable the spring security for authentication using Spring Security

roo> security setup

Created SPRING\_CONFIG\_ROOT/applicationContext-security.xml

Created SRC\_MAIN\_WEBAPP/WEB-INF/views/login.jspx

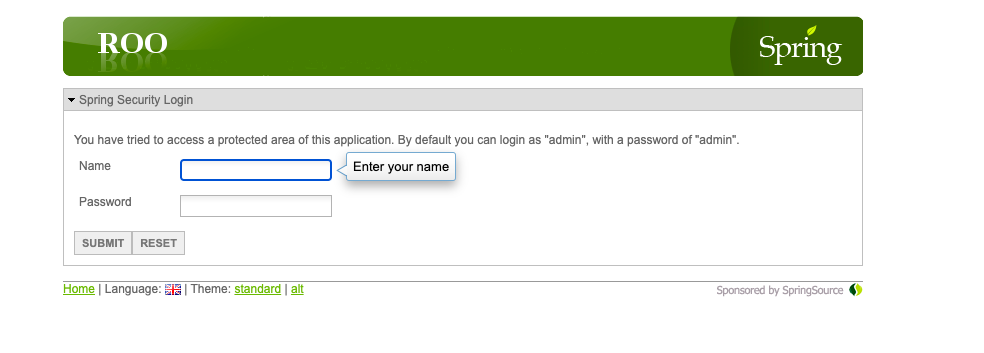
Updated SRC\_MAIN\_WEBAPP/WEB-INF/views/views.xml

Updated ROOT/pom.xml [added property 'spring-security.version' = '3.2.5.RELEASE'; added dependencies org.springframework.security:spring-security-core:${spring-security.version}, org.springframework.security:spring-security-config:${spring-security.version}, org.springframework.security:spring-security-web:${spring-security.version}, org.springframework.security:spring-security-taglibs:${spring-security.version}]

Updated SRC\_MAIN\_WEBAPP/WEB-INF/web.xml

Updated SRC\_MAIN\_WEBAPP/WEB-INF/spring/webmvc-config.xml

roo>



**Note**

The password in the applicationcontext-security.xml is a SHA256 text

<!-- SHA-256 values can be produced using 'echo -n your\_desired\_password | sha256sum' (using normal \*nix environments) -->

<authentication-manager alias="authenticationManager">

<!-- SHA-256 values can be produced using 'echo -n your\_desired\_password | sha256sum' (using normal \*nix environments) -->

<authentication-provider>

<password-encoder hash="sha-256" />

<user-service>

<user name="admin" password="8c6976e5b5410415bde908bd4dee15dfb167a9c873fc4bb8a81f6f2ab448a918" authorities="ROLE\_ADMIN" /> <!--admin-->

<user name="user" password="04f8996da763b7a969b1028ee3007569eaf3a635486ddab211d512c85b9df8fb" authorities="ROLE\_USER" /> <!--user-->

</user-service>

</authentication-provider>

</authentication-manager>

<https://www.dcode.fr/sha256-hash>

## Internationalization Support

Spring Roo adds internationalization support by using the web mvc install language command, which installs a new language in your application. For example, the commands for Spanish and Italian are:

web mvc install language --code es

web mvc install language --code it

## Email Notification

We can add email sending feature into the application. We will use Gmail as our SMTP server to focus on sending email using Roo. Adding email support to an application is done using the following command:

roo> email sender setup --hostServer smtp.gmail.com --username \

<Your email address> --password <Your email password> --port 587 --protocol SMTP

## Socialize Application

This is the age of social media, and social features are commonly added to current applications. It would make sense to add video of Talks. Roo provides support for embedding videos uploaded to YouTube, Vimeo, Viddler, and Google Video, etc.

web mvc embed document

web mvc embed finances

web mvc embed map

web mvc embed photos

web mvc embed stream video

web mvc embed twitter

web mvc embed video