CodeNest Web Application Demo

Prepared by Boris Mechkov

22 May 2017

Prerequisites

Here is what my development environment looked like for this project (prerequisites):

- Windows 8.1 – x64
- JDK8 (jdk1.8.0_73 – 64-Bit) - %JAVA_HOME% variable pointing to it.
- Maven 3.2.3 - %MAVEN_HOME% set and Maven added to the %PATH%
- MySQL Community Server 5.7.18
- Eclipse Neon IDE
- Apache Tomcat 8.0.36

Quick Setup

(all setup files are in the SETUP_FILES folder)

0. Make sure you have all prerequisites covered – JDK8, Maven, MySQL, etc...

1. Prepare the MySQL database
    1. **root/admin** are the default credentials used in Tomcat's DataSource config, so keep in mind
    2. Create the database – run the **CodeNest_MYSQL_DDL.sql** script
    3. Default connection string: **jdbc:mysql://localhost:3306/codenest – change url if needed**

2. Prepare Tomcat8
    1. Copy **customDataSourceFactory.jar AND mysql-connector-java-5.1.42-bin.jar** to **${PATH_TO_TOMCAT_HOME}\lib**
    2. Copy and replace **${PATH_TO_TOMCAT_HOME}\conf\server.xml** with the **server.xml** provided (it adds a new GLOBAL DataSource Resouce pointing to the MySQL database)

3. Build the Web App and Deploy
    1. Navigate to the **demo** folder in the distribution -> run **CMD>mvn package**
    2. Copy **demo.war** from the **demo/target** folder TO **${PATH_TO_TOMCAT_HOME}\webapps**
    3. Start Tomcat (**startup.bat**)
    4. Navigate to: **HTTP://LOCALHOST:8080/DEMO/INDEX.HTML**

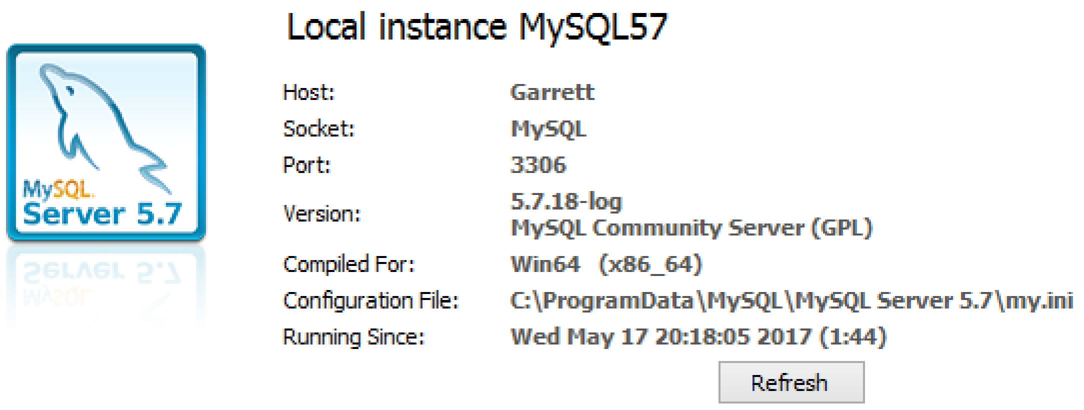
4. **ENJOY!**

Detailed Setup

Database Setup

For this project i am using a MySQL database, running locally. NOTE that to keep it simple, i am using the '**root**' user (password is the default – '**admin**') to interact with the database.

Below is an image of the database environment



Local instance MySQL57

Host:	Garrett
Socket:	MySQL
Port:	3306
Version:	5.7.18-log MySQL Community Server (GPL)
Compiled For:	Win64 (x86_64)
Configuration File:	C:\ProgramData\MySQL\MySQL Server 5.7\my.ini
Running Since:	Wed May 17 20:18:05 2017 (1:44)

I am creating a small database with a single table to handle User accounts. I am attaching a SQL script to speed up its creation, which you can find in the **SETUP_FILES** folder of the distribution (**CodeNest_MYSQL_DDL.sql**)

Web Server Setup

I am using Apache Tomcat 8.0.36 – unzipped distribution to my local machine. I decided to use a container-managed datasource, so you will need to register a global JNDI resource which describes the connection to the MySQL database. I have provided my copy of the **server.xml (SETUP_FILES** folder**)** which can be used to configure Tomcat's datasource – it overrides the **server.xml** in "**${PATH_TO_TOMCAT_HOME}\conf"**. The Resource looks something like this:

```
<Resource auth="Container"
factory="com.mechkov.tomcat.CustomDataSourceFactory.EncryptedDataSourceFactory"
driverClassName="com.mysql.jdbc.Driver" maxActive="100" maxIdle="10"
name="jdbc/mysql" password="824790cf9e17d0d75048f272b197655a"
removeAbandoned="true" removeAbandonedTimeout="30" type="javax.sql.DataSource"
url="jdbc:mysql://localhost:3306/codenest" username="root" validationQuery="select 1"/>
```

NOTE that the above Resource uses encryption for the connection password! The value is just an AES-encrypted string of "admin". Make sure you copy the **customDataSourceFactory.jar** (**SETUP_FILES** folder) to "**${PATH_TO_TOMCAT_HOME}\lib".**

IMPORTANT: Place "mysql-connector-java-5.1.42-bin.jar" in "${PATH_TO_TOMCAT_HOME}\lib" as well. Tomcat needs driver classes to make MySQL connections!

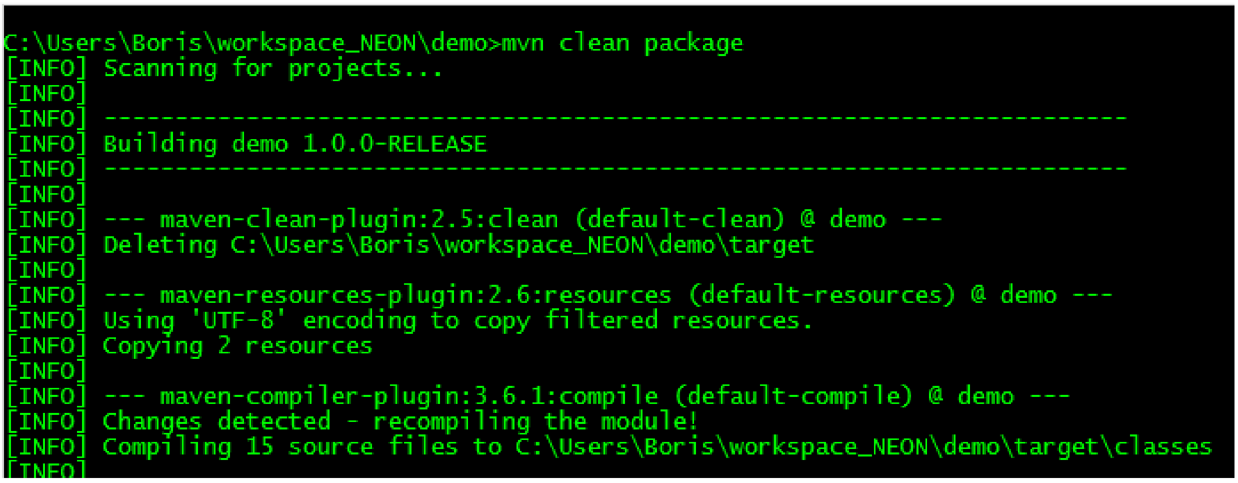
IF you want to keep it simple and for brevity, please use the following Resource instead of the one above (and you don't need the **customDataSourceFactory.jar**):

```
<Resource auth="Container" driverClassName="com.mysql.jdbc.Driver"
factory="org.apache.tomcat.dbcp.dbcp2.BasicDataSourceFactory" maxActive="100"
maxIdle="10" name="jdbc/mysql" password="admin" removeAbandoned="true"
removeAbandonedTimeout="30" type="javax.sql.DataSource"
url="jdbc:mysql://localhost:3306/codenest" username="root" validationQuery="select 1"/>
```

Web Application Setup

The web application is distributed as SOURCE only. To build it, just issue a Maven Package command at the root of the distribution, where the **pom.xml** is located:

CMD>mvn clean package



```
C:\Users\Boris\workspace_NEON\demo>mvn clean package
[INFO] Scanning for projects...
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] Building demo 1.0.0-RELEASE
[INFO] ------------------------------------------------------------------------
[INFO]
[INFO] --- maven-clean-plugin:2.5:clean (default-clean) @ demo ---
[INFO] Deleting C:\Users\Boris\workspace_NEON\demo\target
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ demo ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] Copying 2 resources
[INFO]
[INFO] --- maven-compiler-plugin:3.6.1:compile (default-compile) @ demo ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 15 source files to C:\Users\Boris\workspace_NEON\demo\target\classes
[INFO]
```

Once successfully completed, grab the **demo.war** from the newly created **TARGET** directory and place it in **${PATH_TO_TOMCAT_HOME}\webapps**.

Start Tomcat and once fullyup and running navigate to:
http://{SERVER_URL}:8080/demo/index.html

Please refer to the log folder if there are issues - **${PATH_TO_TOMCAT_HOME}\logs.**
Demo.log is the the application-specific log.