# Lab: Setting Up TomEE with Hibernate and MVC

This **tutorial** provides step-by-step on how to setup TomEE with Hibernate. Additional information will be provided on how to use your custom-made mvc-framework.

Video: <https://softuni.bg/trainings/resources/video/12920/video-screen-23-february-2017-teodor-dimitrov-part-1-java-web-development-basics-january-2017/1536?fbclid=IwAR0gRexfr_iIDGBFPEtJHHtZu3rlx9IH4vpoFKgwWcktyBJP2TJuMkOfVwM>

## Download TomEE

Go to <http://tomee.apache.org/downloads.html> and download **WebProfile.** Extract the web server in a convenient directory. That is all!

## TomEE System Setup

Go to **{TomEE HOME}/conf/system**. Change the following line from

tomee.serialization.class.blacklist = \*

to

tomee.serialization.class.blacklist = -

## TomEE Hibernate Setup

The default **JPA** provider coming up with **TomEE** is **EclipseLink.** However, we want to use **Hibernate.** The way to do this is to import the necessary Hibernate libraries into **{TomEE HOME}/lib.** We have provided them to you to ease the development process. **Here is a** [**link**](https://softuni.bg/downloads/svn/web-development/Jan-2017/Java-Web-Dev-Basics/08.%20Java-Web-Dev-Basics-Workshop-MVC/TomEE%20Libraries.rar)**.**

## TomEE Hibernate Resource Setup

From now on **TomEE** will be the one to take care of our **database connection**. Even more it will handle our **Entity Manager Factory** and return new instances of **Entity Manager**. In order to do this, go to **{TomEE HOME}/conf/tomee.xml.** You need to setup two connections:

* **JTA Managed Connection**
  + It takes care of the **Transactions** and **Entity Manager**
* **JTA Unmanaged Connection**
  + Optional - in Case you want to handle all by yourself

|  |
| --- |
| **tome.xml** |
| **<?xml version="1.0" encoding="UTF-8"?>**  **<tomee>**  **<Resource id="databaseDb" type="DataSource">**  **JdbcDriver com.mysql.jdbc.Driver**  **JdbcUrl jdbc:mysql://localhost:3306/db?createDatabaseIfNotExist=true**  **UserName root**  **Password 1234**  **JtaManaged true**  **</Resource>**    **<Resource id="databaseDbUnmanaged" type="DataSource">**  **JdbcDriver com.mysql.jdbc.Driver**  **JdbcUrl jdbc:mysql://localhost:3306/db?createDatabaseIfNotExist=true**  **UserName root**  **Password 1234**  **JtaManaged false**  **</Resource>**  **</tomee>** |

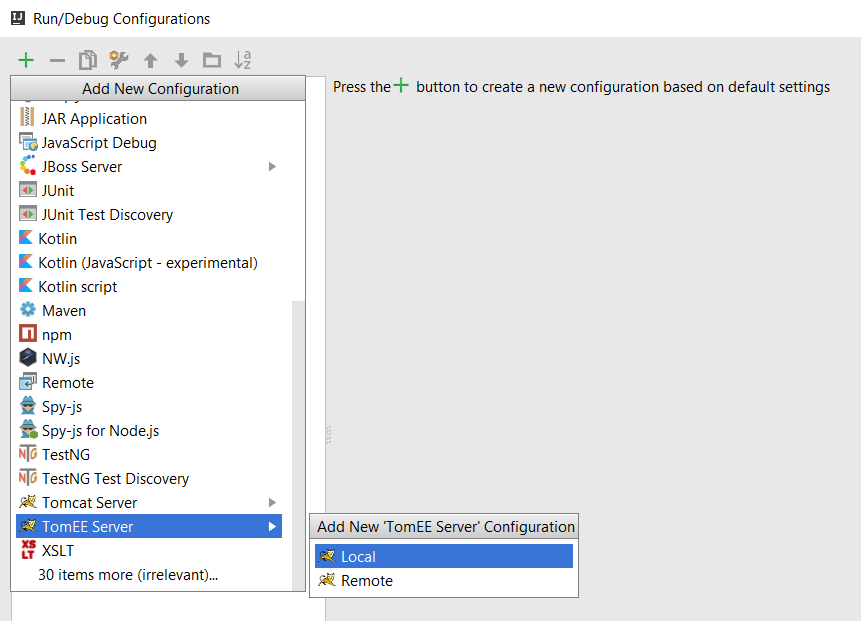
## Persistence Setup

In persistence.xml set a reference to the resources in **TomEE**.

|  |
| --- |
| **persistence.xml** |
| <**persistence version="1.0"  xmlns="http://java.sun.com/xml/ns/persistence"  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xsi:schemaLocation="http://java.sun.com/xml/ns/persistence  http://java.sun.com/xml/ns/persistence/persistence\_1\_0.xsd"**>   <**persistence-unit name="db" transaction-type="JTA"**>  <**provider**>org.hibernate.jpa.HibernatePersistenceProvider</**provider**>  <**jta-data-source**>java:openejb/Resource/databaseDb</**jta-data-source**>  <**non-jta-data-source**>java:openejb/Resource/databaseDbUnmanaged</**non-jta-data-source**>   <**properties**>  <**property name="hibernate.dialect" value="org.hibernate.dialect.MySQL5Dialect"**/>  <**property name="hibernate.hbm2ddl.auto" value="update"**/>  <**property name="hibernate.show\_sql" value="true"**/>  <**property name="tomee.jpa.factory.lazy" value="true"** />  </**properties**>   </**persistence-unit**>  </**persistence**> |

## IntelliJ TomEE Setup

Open IntelliJ and add TomEE like it is shown.



## Project Setup

This is an example pom.xml

|  |
| --- |
| **pom.xml** |
| <**dependencies**>  *<!-- JavaEE -->* <**dependency**>  <**groupId**>javax</**groupId**>  <**artifactId**>javaee-web-api</**artifactId**>  <**version**>7.0</**version**>  </**dependency**>  *<!-- MySQL -->* <**dependency**>  <**groupId**>mysql</**groupId**>  <**artifactId**>mysql-connector-java</**artifactId**>  <**version**>5.1.38</**version**>  </**dependency**>  *<!--Hibernate -->* <**dependency**>  <**groupId**>org.hibernate</**groupId**>  <**artifactId**>hibernate-core</**artifactId**>  <**version**>5.2.5.Final</**version**>  </**dependency**>  *<!-- JSTL -->* <**dependency**>  <**groupId**>jstl</**groupId**>  <**artifactId**>jstl</**artifactId**>  <**version**>1.2</**version**>  </**dependency**>  </**dependencies**>  <**properties**>  <**maven.compiler.source**>1.8</**maven.compiler.source**>  <**maven.compiler.target**>1.8</**maven.compiler.target**> </**properties**> |

## Repository Setup

@PersistanceContext will ensure a dependency injection to the Entity Manager

|  |
| --- |
| **UserRepository.java** |
| @Stateless **public class** UserRepositoryImpl **implements** UserRepository {   @PersistenceContext  **private** EntityManager **entityManager**;   @Override  **public void** create(User user) {  **this**.**entityManager**.persist(user);  } } |

## Controller Setup

@Inject will ensure a dependency injection to any other object.

|  |
| --- |
| **UserRepository.java** |
| @Controller **public class** UserController {   @Inject  **private** UserRepository **userRepository** = **new** UserRepositoryImpl();   @GetMapping(**"/register"**)  **public** String register() {  **return "register"**;  } } |