# 第八周周报

## 摘要

本周我们继续开发了服务器端的后台管理系统，我们分别尝试了django和java web两种框架进行构建。由于正好小学期学习了一些java EE相关的知识，我们活学活用，将课程中学到的entity beans、session beans等内容与大创的项目结合，在这一周里开发了一下项目网站的后台管理系统。

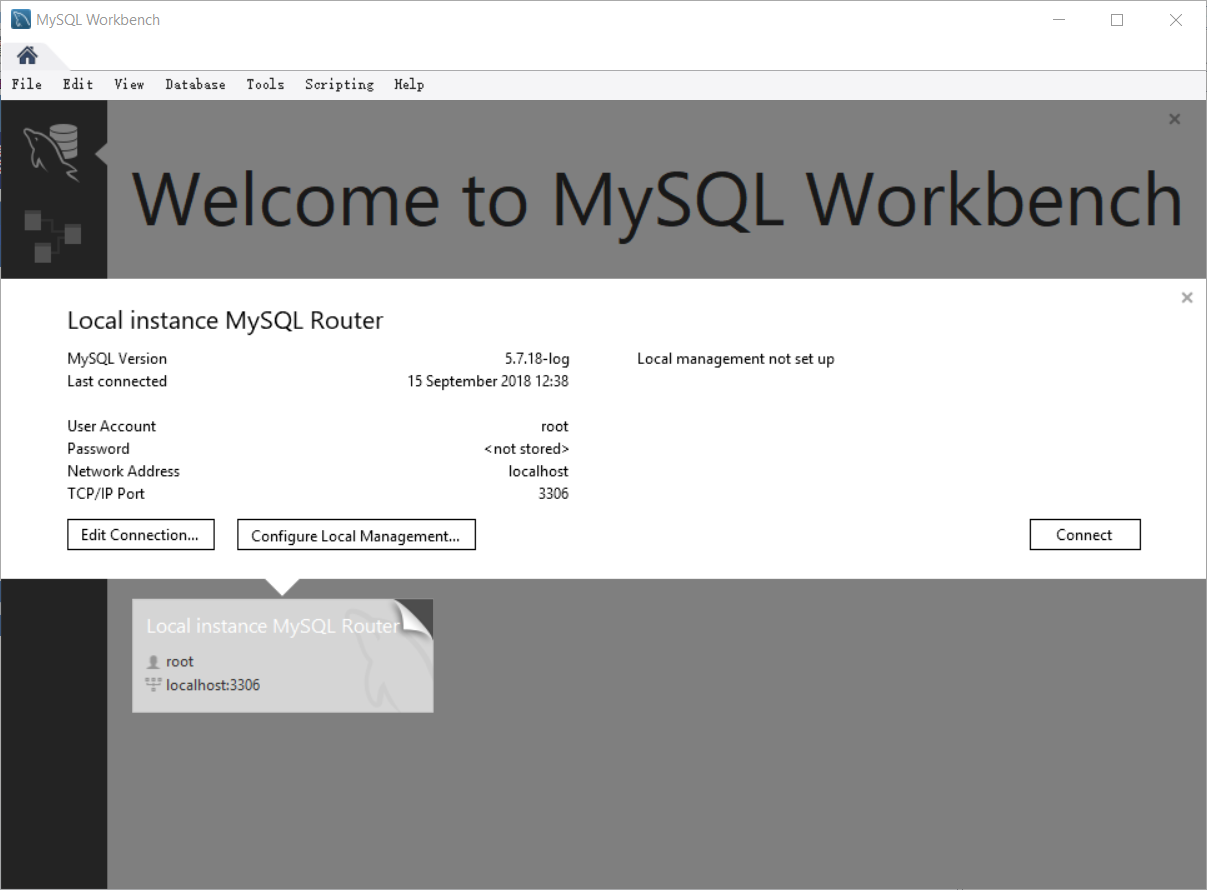
以下是本周的学习步骤。

# 使用Java Web开发网站后台管理系统

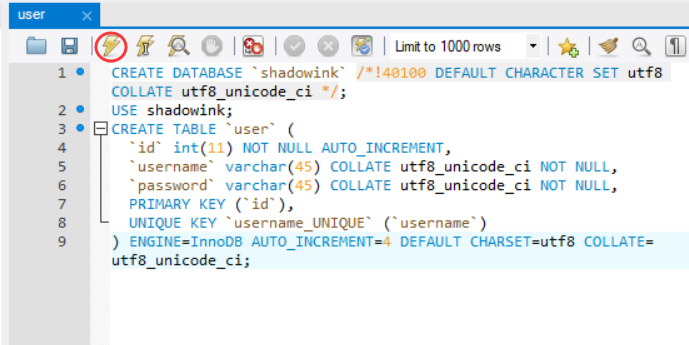
开发环境：Windows 10， NetBeans IDE 8.2，Java SDK 1.8，MySQL Workbench 6.3。

## 建立数据库

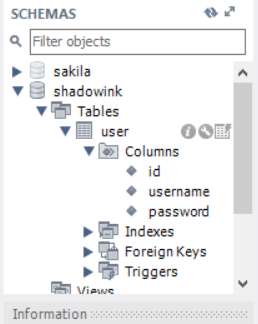
1. 首先打开MySQL Workbench，建立一个连接。



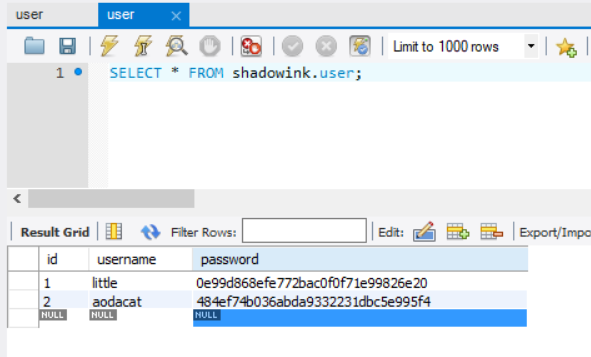
1. 输入SQL语句建立数据库并建表。点击红圈内的闪电按钮运行SQL语句。



1. 在左下角的工具栏中可以看到我们已经建立好了一个名为Shadowink的数据库。

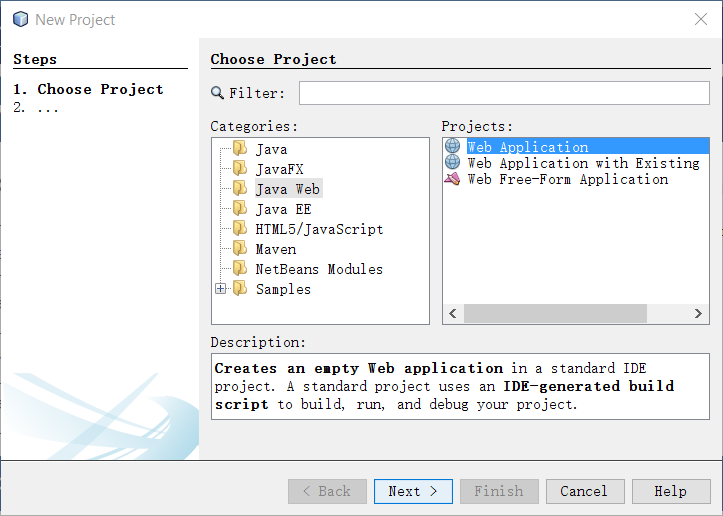


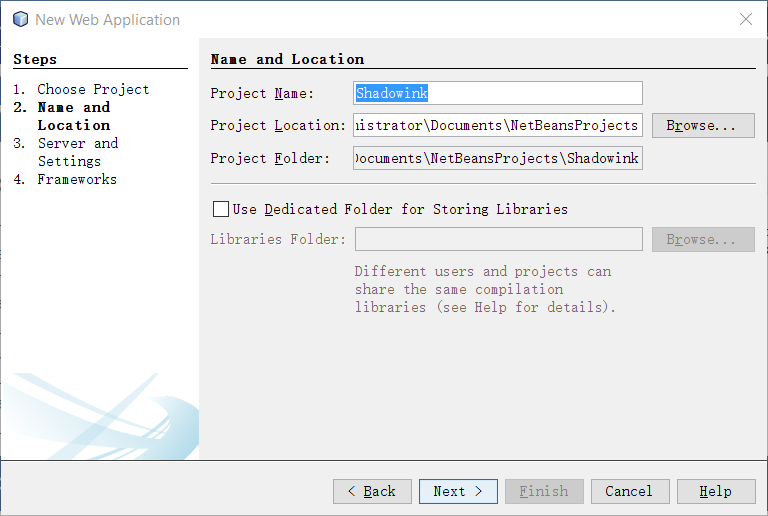
1. 在user上双击，可以查看表的信息，并可以手动输入一些数据用于测试。

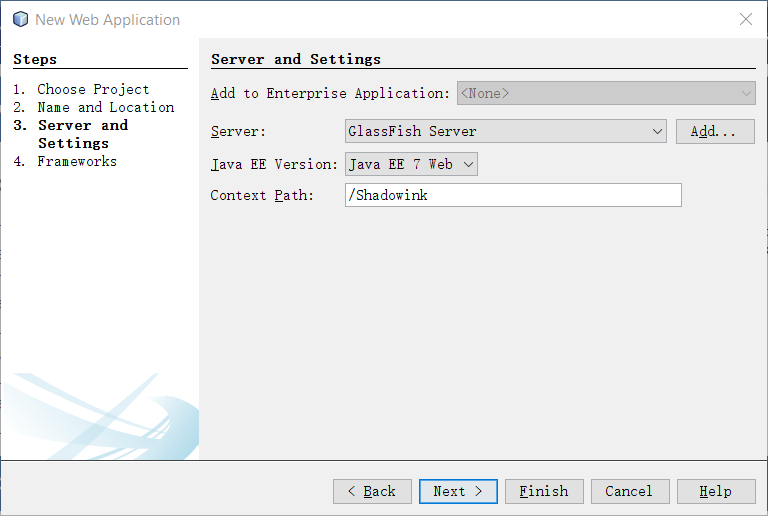


## 建立Java Web项目

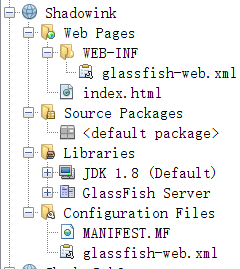
1. 在NetBeans中建立Web Application项目，起名为ShadowInk，选择安装的Glassfish或Tomcat服务器，然后选择Java EE 7框架，不使用第三方框架。



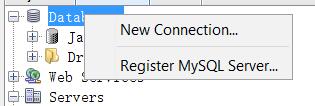


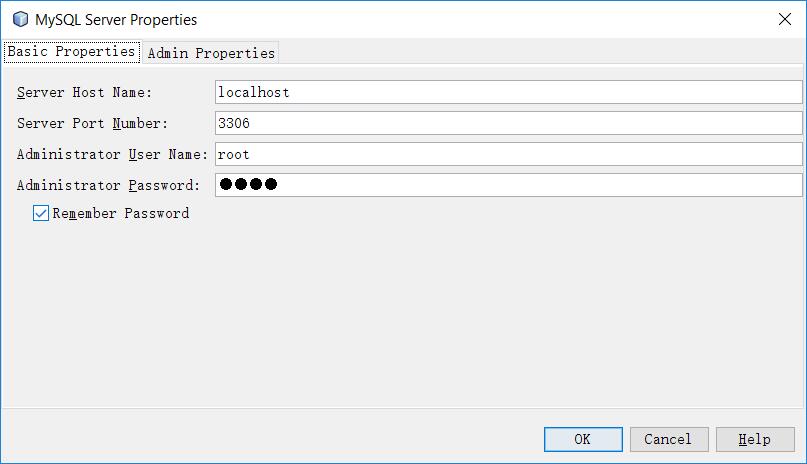


1. 建立后的项目结构如图所示：

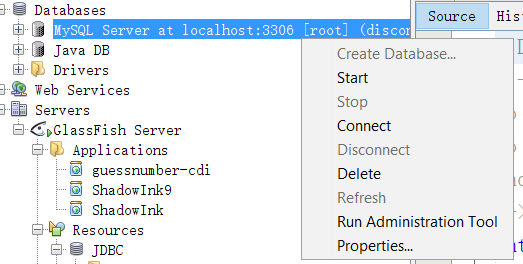


1. 在左边的标签栏中选择“服务”，然后在服务器中选择“注册MySQL服务器”，设置你的MySQL服务器的端口地址，用户名和密码：

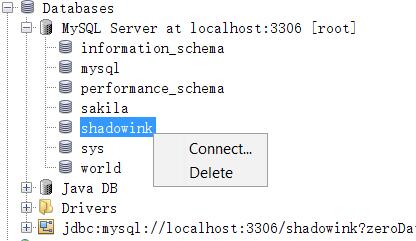




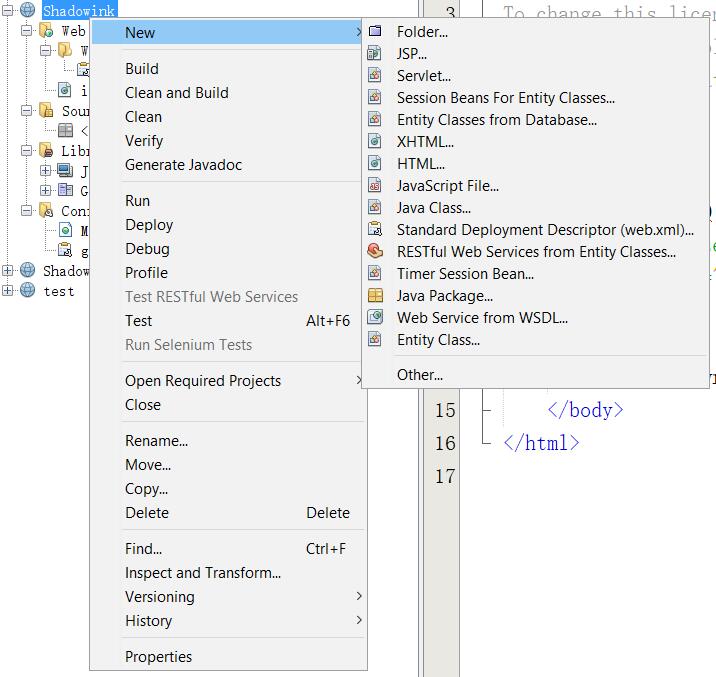
1. 在建立后的服务器上右键选择“连接”，如果服务器配置没有问题，就应当可以连接上MySQL的服务器。



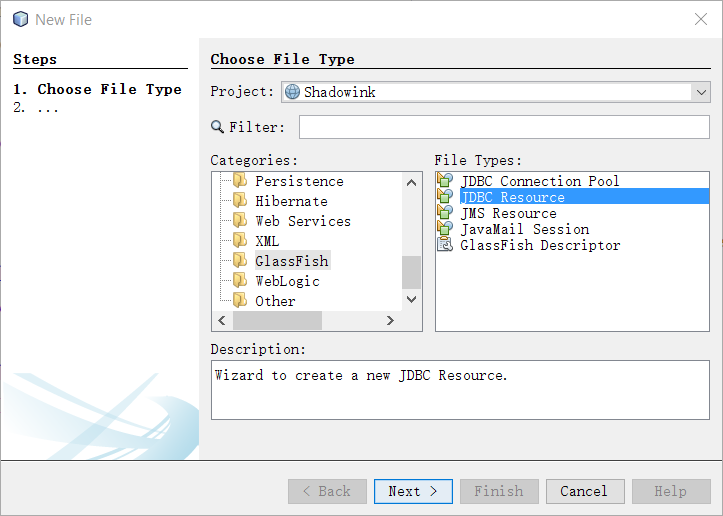
1. 连接上的服务器中选择我们需要的数据库，右键再点击“连接”。连接后的数据库如下图所示：



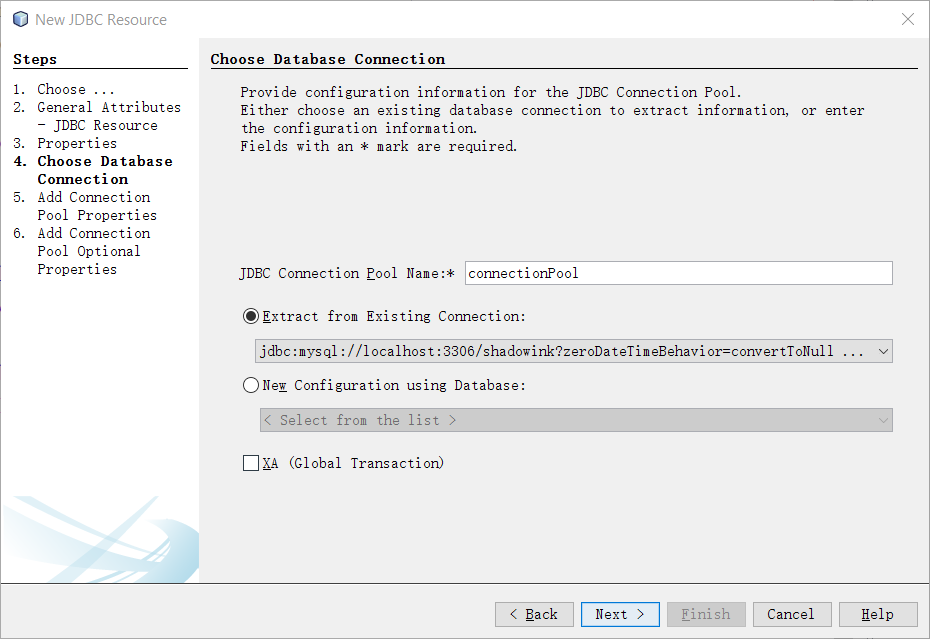
1. 在NetBeans中按Ctrl+F5，或者直接在项目上右击并选择“新建...”，就可以手动向项目中添加文件。



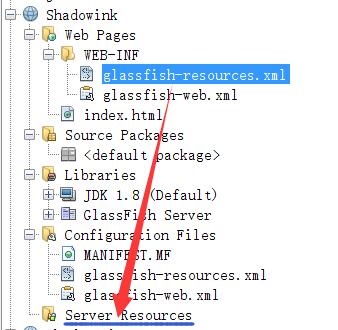
依次在我们的项目中添加一个JDBC连接池和一个JDBC资源。



建立的过程中的命名等都可以直接使用默认的；在选择数据库的步骤中一定要选择刚才连接上的shadowink数据库，如下图所示：



1. 很重要的一步！把WEB-INF中生成了的glassfish-resources.xml拖到Server Resources目录中。

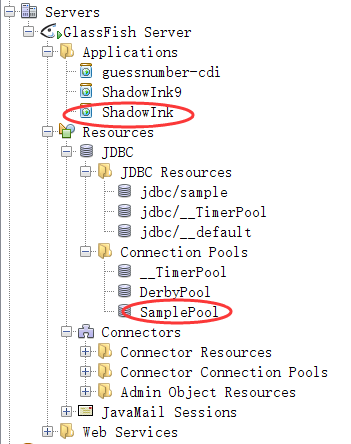


这样是为了让IDE根据我们的JDBC资源配置文件生成相应的JDBC资源。

1. 在项目上右键选择“Deploy”（中文版为部署），等待项目部署完成。

每当项目的配置文件发生修改（如修改文件名、添加依赖、修改编码格式等）后，最好都要重新部署一下项目，否则可能会出现奇怪的问题。

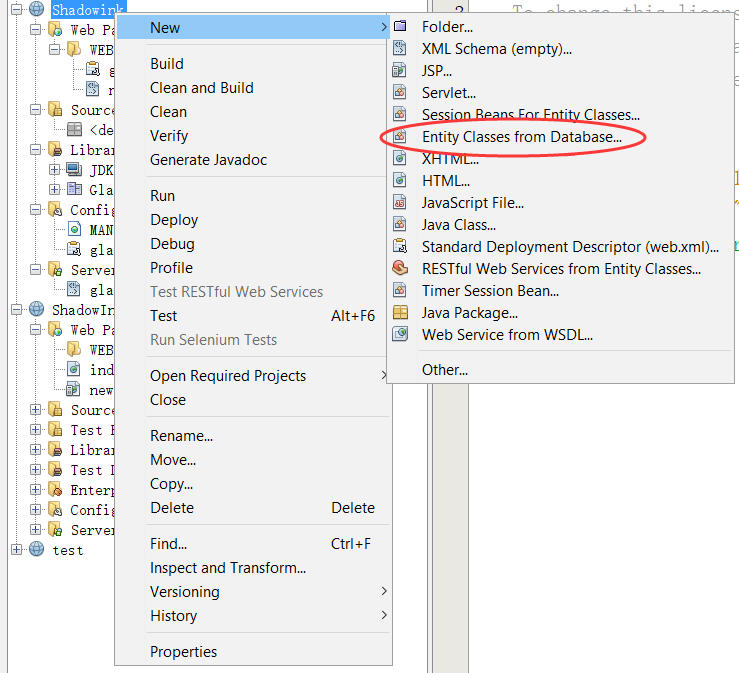
部署完成后可以在service中看到我们的项目已经在服务器上准备就绪：

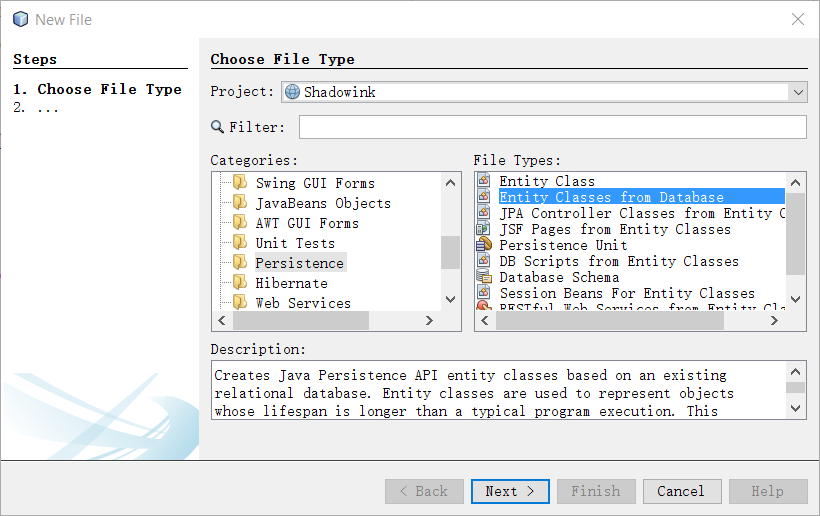


## Entity Beans

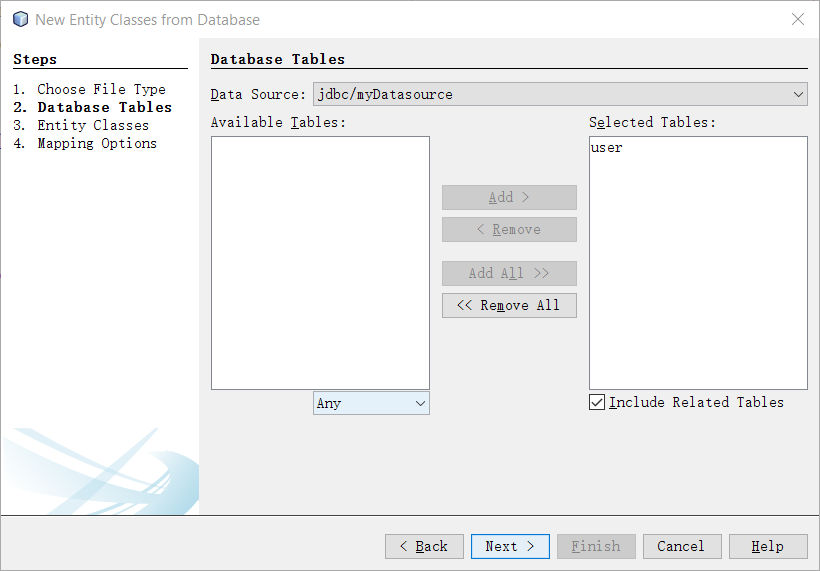
1. 在项目中加入一个“从数据库创建实体类”对象，如下图红圈所示。

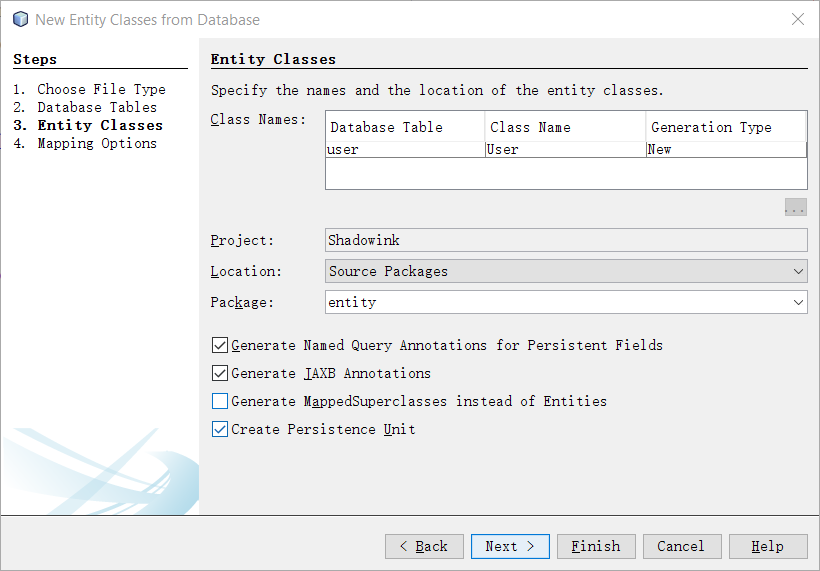
如果找不到这个选项，它有可能在Other -> Persistence 中。

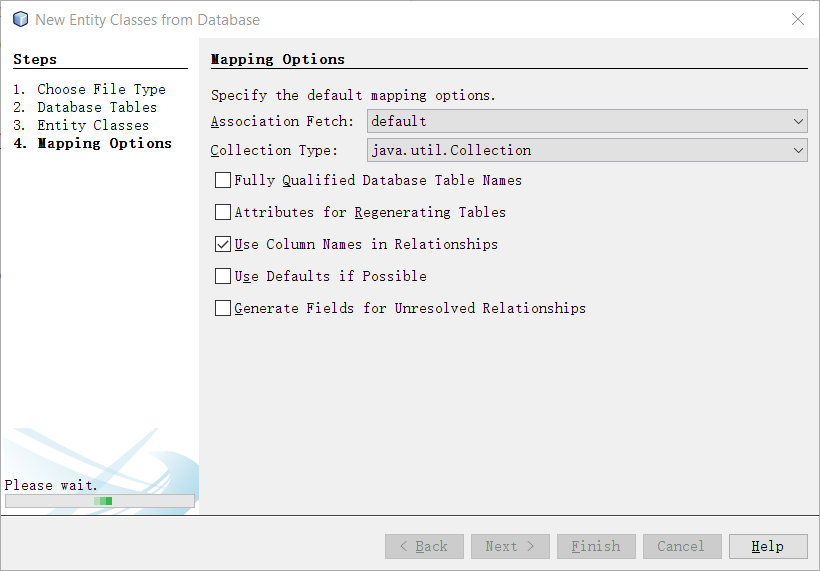




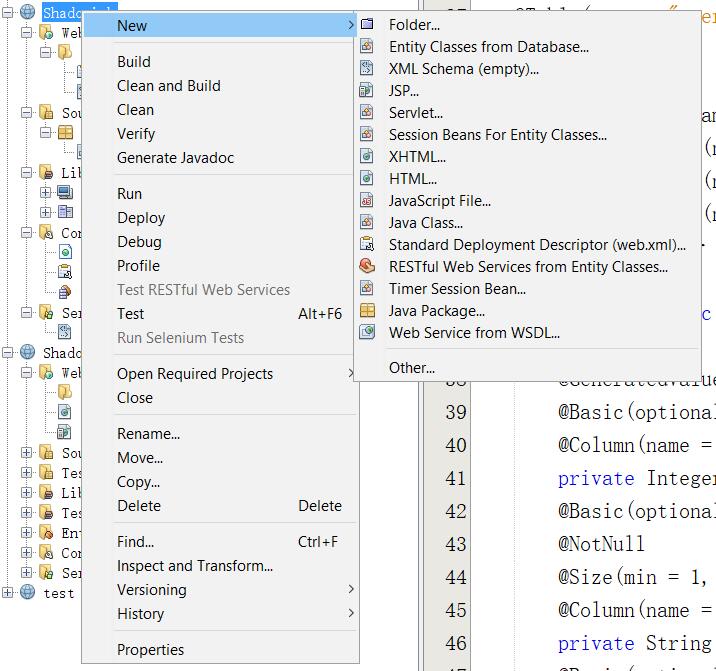
1. 名字直接选择默认不变，选择表格时选择我们刚才建立的JVBC资源，包名设置为entity，然后选择继续。



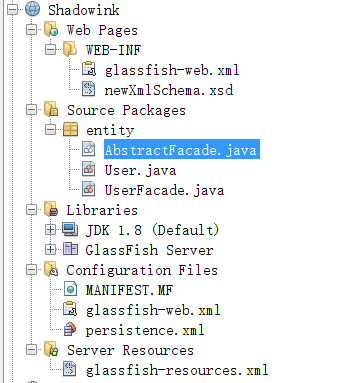




1. 类似的方法再建立一个Session Beans for entity class。



1. 最终的项目结构如下图所示：



## 读写数据库

1. 现在Java EE自动通过我们的数据库构建了一组实体类，其中UserFacade就是一个Session Bean实体，而User是一个Entity Bean实体。在UserFacade中自动封装了对数据库的增删查改等操作。
2. 因为数据库中的行已经被映射到了java中的User对象，我们可以通过以下代码对数据库进行插入：

UserFacade userFacade;

User new\_user = new User((int)(Math.random()\*100),username,password);

userFacade.create(new\_user);

1. 编写后的注册页面Servlet处理逻辑如下：

@WebServlet(loadOnStartup = 1,urlPatterns = {"/register"})

public class NewServlet extends HttpServlet {

@EJB

private UserFacade userFacade;

@Override

public void init() throws ServletException {

// store category list in servlet context

getServletContext().setAttribute("users", userFacade.findAll());

}

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

System.out.println(request.getContextPath());

if(request.getRequestURI().equals("/ShadowInk9/register") ){

String username = request.getParameter("username");

String password = request.getParameter("password");

User new\_user = new User((int)(Math.random()\*100),username,password);

userFacade.create(new\_user);

System.out.println(request);

response.sendRedirect("/ShadowInk9/newjsp.jsp");

System.out.println(request);

}

}

## 编写界面

1. 编写jsp代码。其中关键的数据库操作部分如下：

<c:forEach var="user" items="${users}">

id: ${user.id} username: ${user.username} password:${user.password}

</c:forEach>

1. 为了快速地获得一个能看的界面，我们使用bootstrap前端库来进行一些界面美化。

本周暂时做到这里，以上先作为一个整理，下一周准备继续。