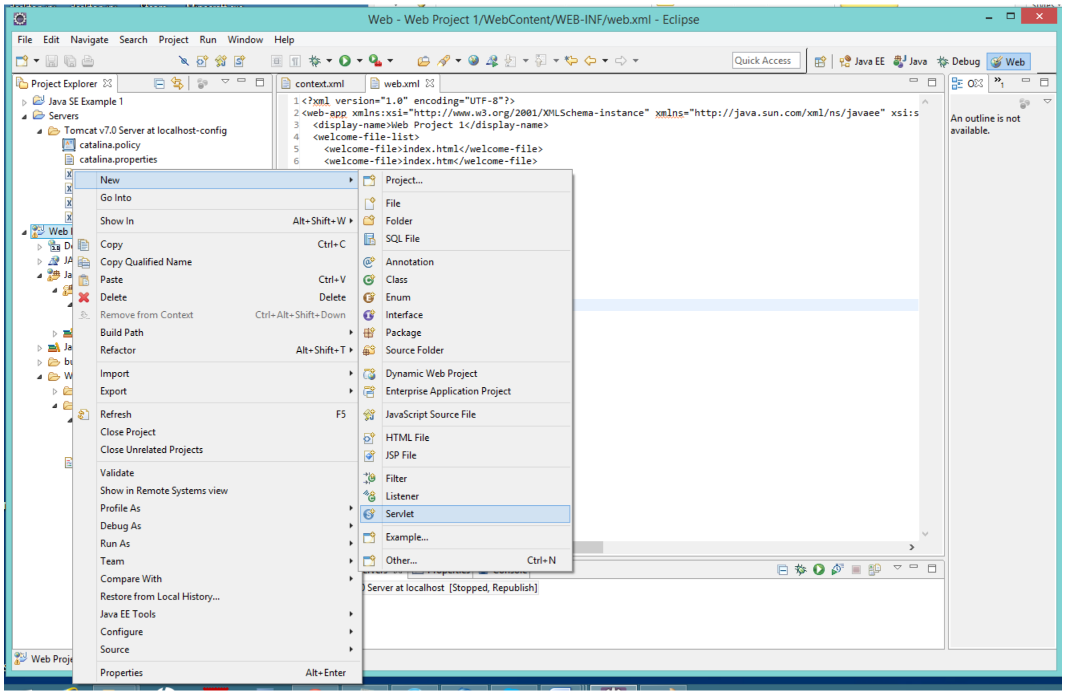
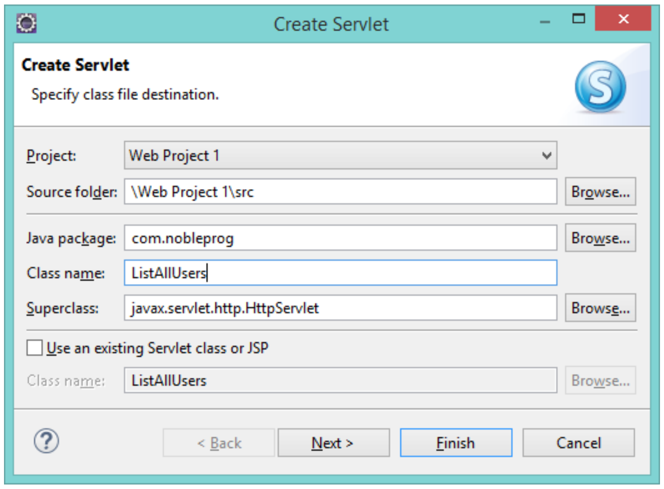
Create a Servlet that uses JNDI

Open Eclipse if it is not already open and create a Servlet. To create a Servlet right click on the Web Project and choose new/Servlet from the menu as shown below



```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee" xsi:s
  <display-name>Web Project 1</display-name>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
    <welcome-file>index.htm</welcome-file>
```

Call the servlet ListAllUsers and put it into the com.nobleprog package



Create Servlet

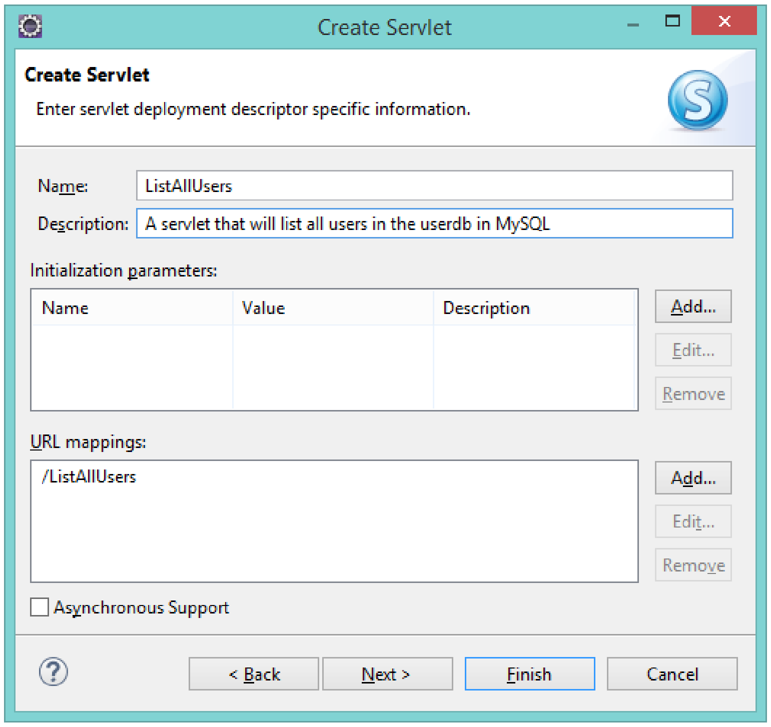
Specify class file destination.

Project:	Web Project 1
Source folder:	\Web Project 1\src
Java package:	com.nobleprog
Class name:	ListAllUsers
Superclass:	javax.servlet.http.HttpServlet

☐ Use an existing Servlet class or JSP

Class name: ListAllUsers

Click on Next and go to the next Dialog. Put in an appropriate description and click Next.



Create Servlet

Enter servlet deployment descriptor specific information.

Name: ListAllUsers

Description: A servlet that will list all users in the userdb in MySQL

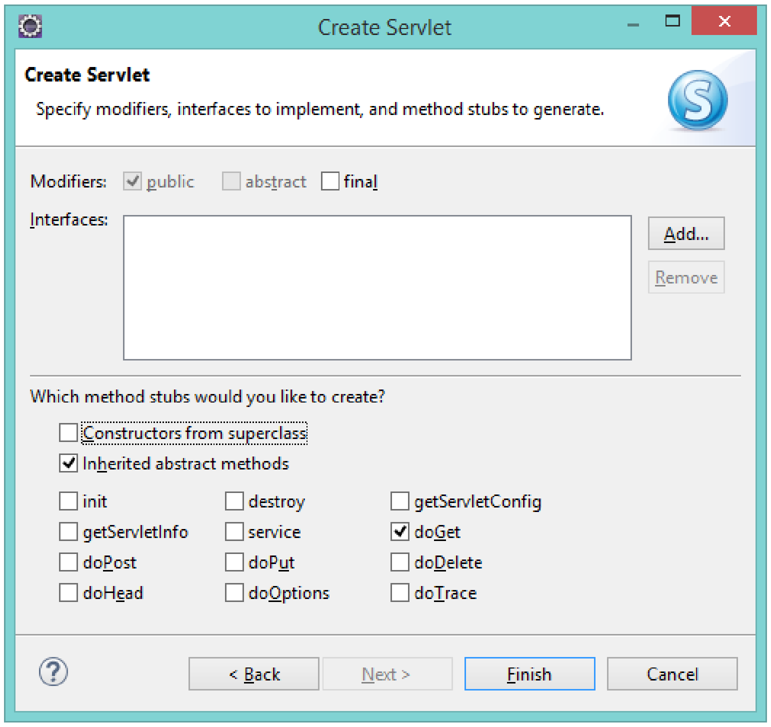
Initialization parameters:

Name	Value	Description

URL mappings:

/ListAllUsers

☐ Asynchronous Support



Check off only the Inherited abstract methods and doGet stubs and click Finish. This will create the servlet.

Create Servlet

Specify modifiers, interfaces to implement, and method stubs to generate.

Modifiers: ☑ public ☐ abstract ☐ final

Interfaces:

Which method stubs would you like to create?

☐ Constructors from superclass

☑ Inherited abstract methods

☐ init	☐ destroy	☐ getServletConfig
☐ getServletInfo	☐ service	☑ doGet
☐ doPost	☐ doPut	☐ doDelete
☐ doHead	☐ doOptions	☐ doTrace

Replacement Code for the Servlet

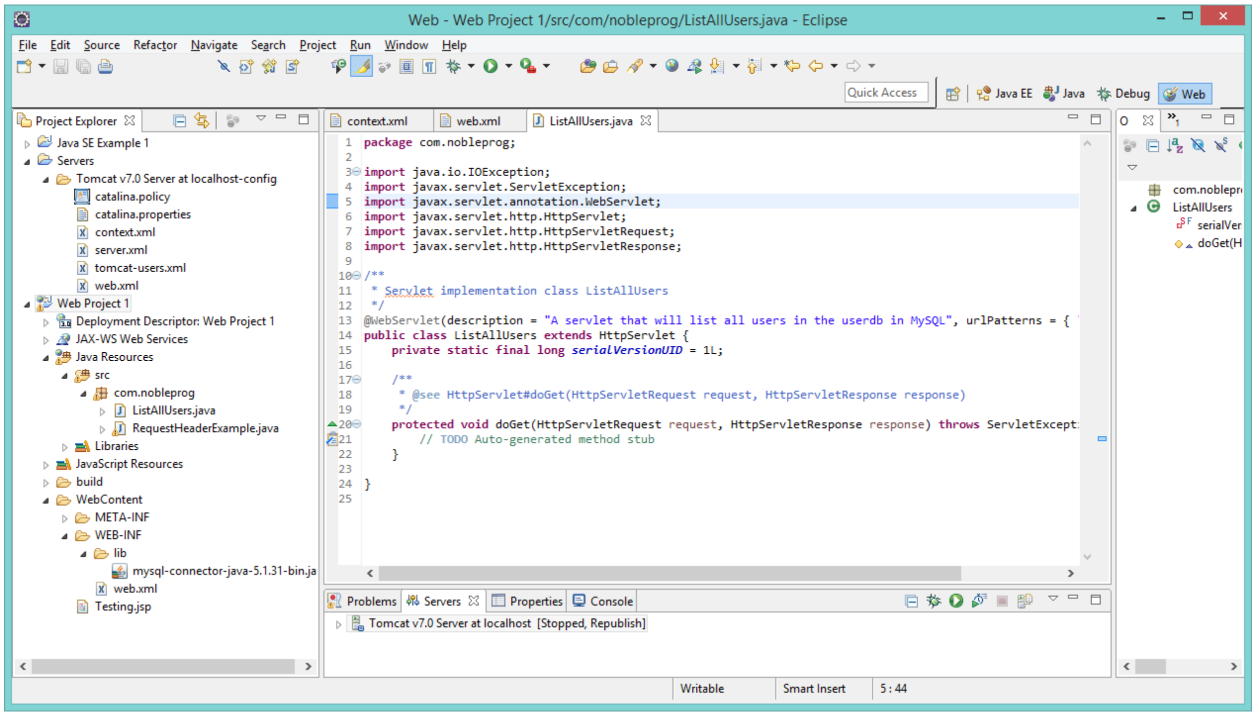
This is the code that will be used to replace code in the imports and doGet method of the Servlet.

```
package com.nobleprog;

import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.naming.Context;
import javax.naming.InitialContext;
import javax.naming.NamingException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.sql.DataSource;


/**
 * Servlet implementation class ListAllUsers
 */
@WebServlet(description = "A servlet that will list all users in the userdb in MySQL",
urlPatterns = { "/ListAllUsers" })
public class ListAllUsers extends HttpServlet {
       private static final long serialVersionUID = 1L;

    protected void doGet(HttpServletRequest request,
            HttpServletResponse response) throws ServletException, IOException {
        PrintWriter writer = response.getWriter();
        try {
            Context initContext = new InitialContext();
            Context envContext = (Context) initContext.lookup("java:comp/env");
            DataSource ds = (DataSource) envContext.lookup("jdbc/userdb");
            Connection conn = ds.getConnection();

            Statement statement = conn.createStatement();
            String sqlSelect = "select username, email from users";
            ResultSet results = statement.executeQuery(sqlSelect);

            int count = 1;
            while (results.next()) {
               String name = results.getString("username");
               String email = results.getString("email");
               writer.println("User" + count++ + " is " + name + " the user's email is " +
email + "<br/>");
            }
        } catch (NamingException ex) {
            System.err.println(ex);
        } catch (SQLException ex) {
            System.err.println(ex);
        }
    }

}
```

Put New Code in Servlet

The servlet class after creation is as shown below. Using the example code on the last page replace the import statements with the ones from the example code. Make sure that you have expanded the imports to show all of them before you erase and replace them (otherwise you will get duplicate imports which is an error).

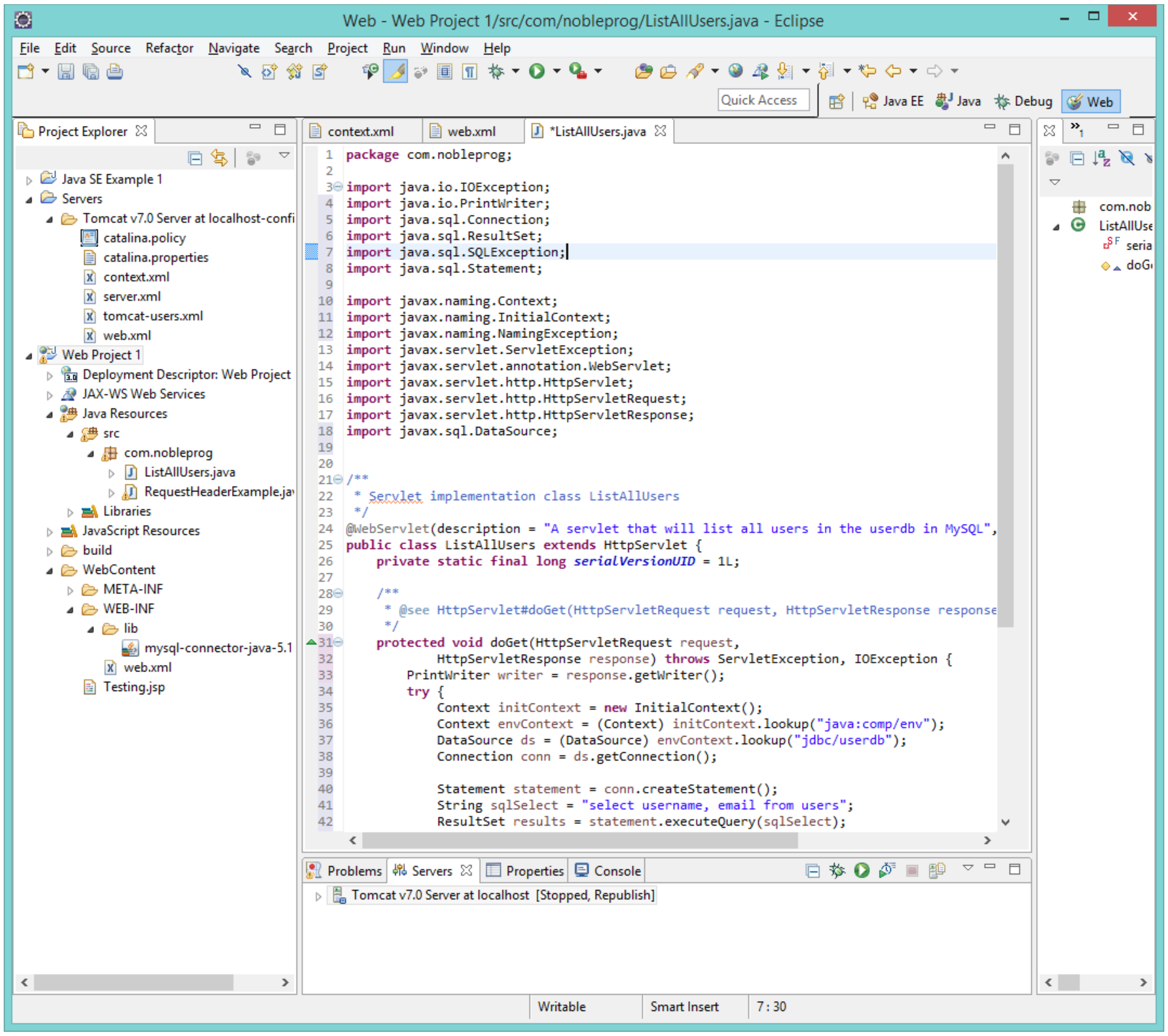
Also replace the doGet method with the new one given below. If you have named the class and the package with the exact name of the class and package in the code below then you can replace everything. If not you must replace the import and the doGet method only.



```
package com.nobleprog;

import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 * Servlet implementation class ListAllUsers
 */
@WebServlet(description = "A servlet that will list all users in the userdb in MySQL", urlPatterns = {
public class ListAllUsers extends HttpServlet {
	private static final long serialVersionUID = 1L;

	/**
	 * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
	 */
	protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletExcept
		// TODO Auto-generated method stub
	}

}
```

Tomcat v7.0 Server at localhost [Stopped, Republish]

You should have no errors when done and the code should look like the image below (partial image of the code is shown in Eclipse):



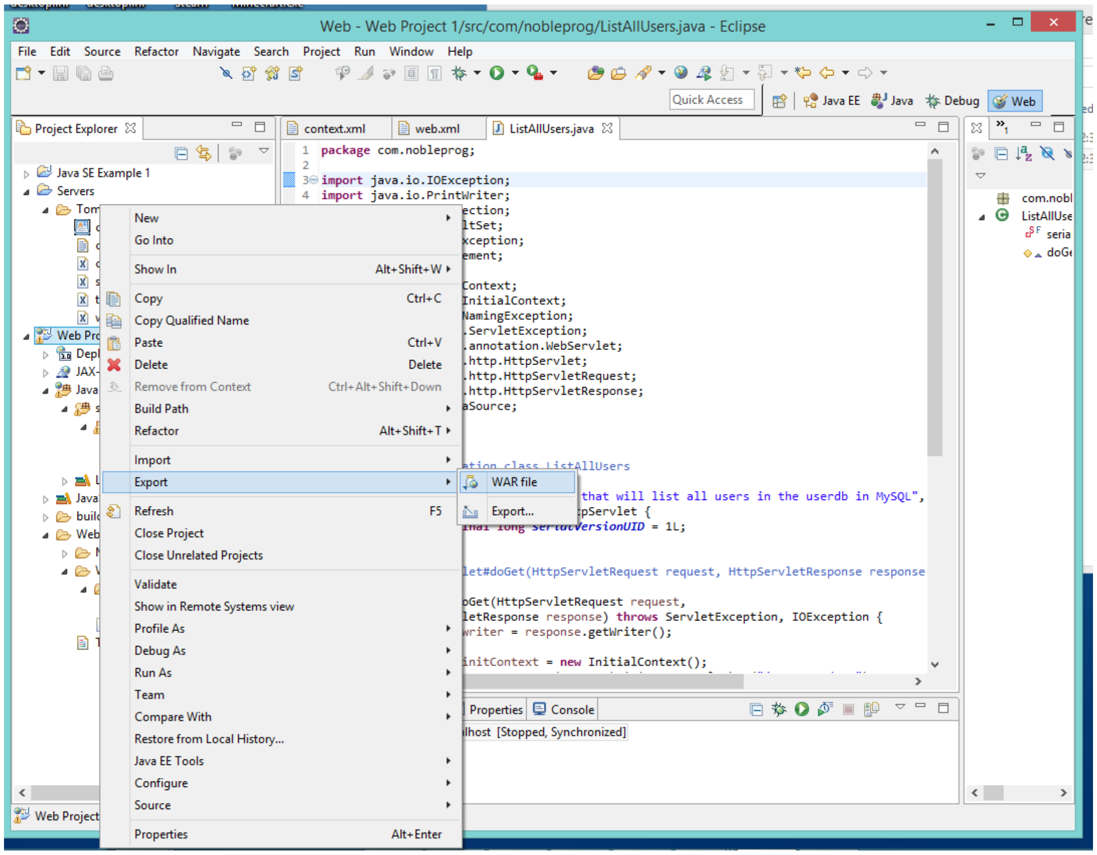
```java
package com.nobleprog;

import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.naming.Context;
import javax.naming.InitialContext;
import javax.naming.NamingException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.sql.DataSource;


/**
 * Servlet implementation class ListAllUsers
 */
@WebServlet(description = "A servlet that will list all users in the userdb in MySQL",
public class ListAllUsers extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response
     */
    protected void doGet(HttpServletRequest request,
            HttpServletResponse response) throws ServletException, IOException {
        PrintWriter writer = response.getWriter();
        try {
            Context initContext = new InitialContext();
            Context envContext = (Context) initContext.lookup("java:comp/env");
            DataSource ds = (DataSource) envContext.lookup("jdbc/userdb");
            Connection conn = ds.getConnection();

            Statement statement = conn.createStatement();
            String sqlSelect = "select username, email from users";
            ResultSet results = statement.executeQuery(sqlSelect);
```

Now right click on the Servlet code and choose Run As/Run on Server. This will allow you to choose Tomcat 7 that was installed on Eclipse in a prior exercise. Tomcat will run and display the file results which should look similar to the following (depending upon what was put into the user table in MySQL).

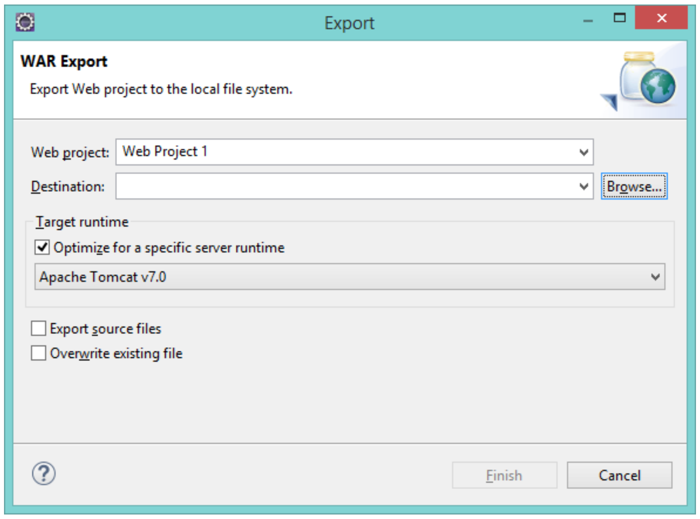
User1 is Harry the user's email is harry@someplace.net
User2 is George the user's email is george@someplace.net

Create a WAR and Deploy to Tomcat

Right click on the Web project and choose Export/WAR File (this option should come up since we created WAR files in prior exercises. If it does not come up choose Export… and find WAR file in the Dialog that appears, maybe under Web).



```
package com.nobleprog;

import java.io.IOException;
import java.io.PrintWriter;
ection;
ltSet;
xception;
ement;

Context;
InitialContext;
NamingException;
.ServletException;
.annotation.WebServlet;
.http.HttpServlet;
.http.HttpServletRequest;
.http.HttpServletResponse;
aSource;


ation class ListAllUsers
that will list all users in the userdb in MySQL",
pServlet {
inal long serialVersionUID = 1L;

let#doGet(HttpServletRequest request, HttpServletResponse response

oGet(HttpServletRequest request,
letResponse response) throws ServletException, IOException {
writer = response.getWriter();

initContext = new InitialContext();
```



Web project: Web Project 1

Destination:

Target runtime

☑ Optimize for a specific server runtime

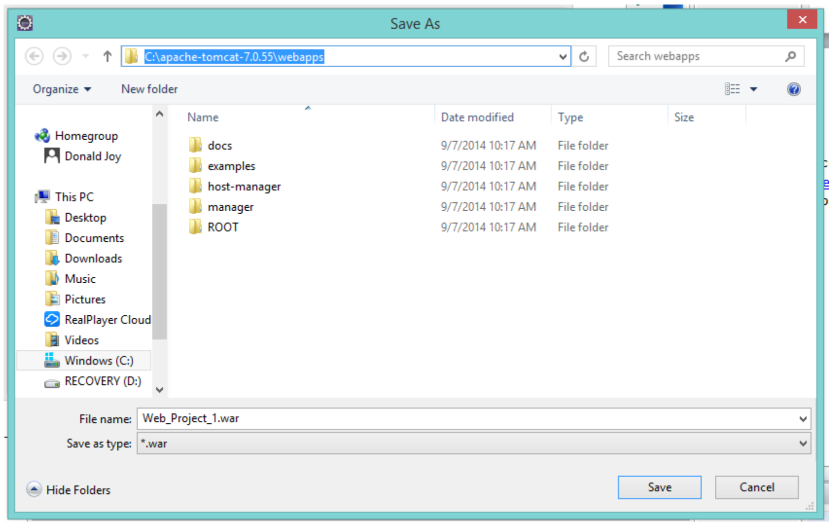
Apache Tomcat v7.0

☐ Export source files

☐ Overwrite existing file

Choose the destination for the WAR file, which should be the Tomcat webapps directory by clicking Browse.

Note that to get the exact same results that occurred in Eclipse, that is the URL of the servlet being: http://localhost:8080/Web_Project_1/ListAllUsers (underscores between the words giving Web_Project_1) the name of the WAR must be changed to Web_Project_1. This occurs because the name of the Web Application on Tomcat is the name of the WAR.


Save As

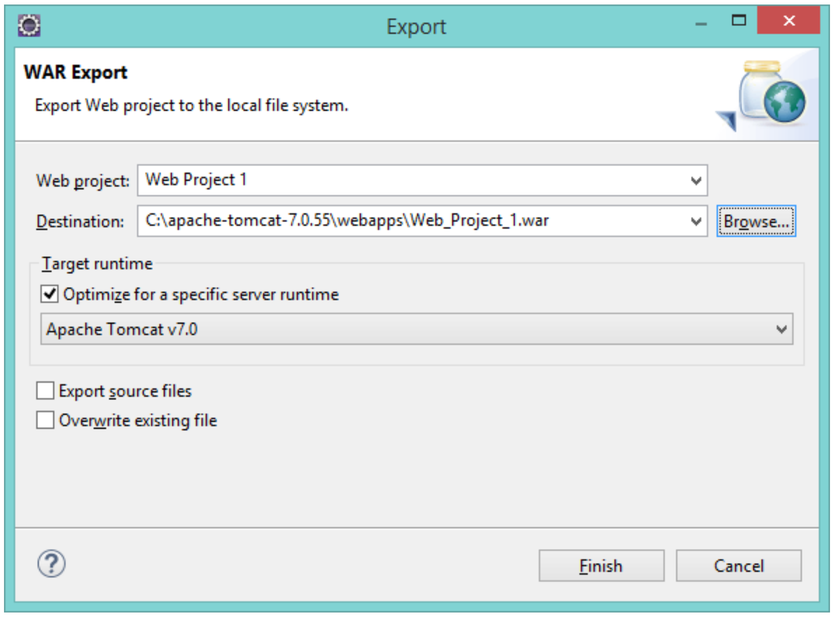
C:\apache-tomcat-7.0.55\webapps

Name	Date modified	Type	Size
docs	9/7/2014 10:17 AM	File folder	
examples	9/7/2014 10:17 AM	File folder	
host-manager	9/7/2014 10:17 AM	File folder	
manager	9/7/2014 10:17 AM	File folder	
ROOT	9/7/2014 10:17 AM	File folder	

File name: Web_Project_1.war

Save as type: *.war

This gives the Destinations shown below:


Export

WAR Export

Export Web project to the local file system.

Web project: Web Project 1

Destination: C:\apache-tomcat-7.0.55\webapps\Web_Project_1.war

Target runtime

☑ Optimize for a specific server runtime

Apache Tomcat v7.0

☐ Export source files

☐ Overwrite existing file

Now click Finish and the WAR file will deploy to Tomcat. In server.xml in the Host element autoDeploy has been set to True so the WAR will be deployed when Tomcat starts or when the file changes if Tomcat is already started.

Stop Tomcat in Eclipse if it is running. Start Tomcat with the startup.bat file in the Tomcat installations bin directory. Try the URL given when you ran the ListAllUsers file within Tomcat in a browser such as Firefox, IE, or Chrome. The WAR should deploy and the results should be the same as in Eclipse.