**Java web development with Eclipse WTP – Tutorial**

[*http://www.vogella.com/tutorials/EclipseWTP/article.html*](http://www.vogella.com/tutorials/EclipseWTP/article.html)

**Lars Vogel**

Version 2.5

Copyright © 2008, 2009, 2010, 2011, 2012, 2013, 2014 vogella GmbH

08.12.2014

**Eclipse Web Tool Platform (WTP)**

This tutorial describes the development of servlets with Eclipse WTP. This tutorial is based on Eclipse 4.4 (Luna) and Tomcat 7.0 and JDK 1.6.

**Table of Contents**

[**1. Eclipse Web Tool Platform**](http://www.vogella.com/tutorials/EclipseWTP/article.html#overview)

[**2. Tomcat Installation**](http://www.vogella.com/tutorials/EclipseWTP/article.html#installationtomcat)

[**3. Installation of WTP**](http://www.vogella.com/tutorials/EclipseWTP/article.html#installation)

[**4. WTP Configuration**](http://www.vogella.com/tutorials/EclipseWTP/article.html#configuration)

[**4.1. Setting up runtime environments**](http://www.vogella.com/tutorials/EclipseWTP/article.html#configuration_runtime)

[**4.2. Server**](http://www.vogella.com/tutorials/EclipseWTP/article.html#serverhandling)

[**5. Exercise: Create a servlet**](http://www.vogella.com/tutorials/EclipseWTP/article.html#servlets)

[**5.1. Target**](http://www.vogella.com/tutorials/EclipseWTP/article.html#servlets_target)

[**5.2. Create web project**](http://www.vogella.com/tutorials/EclipseWTP/article.html#servlets_project)

[**5.3. Creating Data Access Object**](http://www.vogella.com/tutorials/EclipseWTP/article.html#servlets_dao)

[**5.4. Creating the Servlet**](http://www.vogella.com/tutorials/EclipseWTP/article.html#servlets_createservlet)

[**5.5. Deployment of the servlet**](http://www.vogella.com/tutorials/EclipseWTP/article.html#servlets_runservlet)

[**5.6. Validate the deployment**](http://www.vogella.com/tutorials/EclipseWTP/article.html#servlets_validatedeployment)

[**6. Web Archive - How to create a war file from Eclipse**](http://www.vogella.com/tutorials/EclipseWTP/article.html#wtp_war)

**1. Eclipse Web Tool Platform**

The *Eclipse WTP* project provides tools for developing standard Java web applications and Java EE applications. Typical web artifacts in a Java environment are HTML pages, XML files, webservices, servlets and JSPs. Eclipse WTP simplifies the creation these web artifacts and provides runtime environments in which these artifacts can be deployed, started and debugged.

In Eclipse WTP you create *Dynamic Web Projects*. These projects provide the necessary functionality to run, debug and deploy Java web applications.

Eclipse WTP supports all major web containers. This includes Jetty and Apache Tomcat as well as most Java EE application server. In this tutorial we use Apache Tomcat as the web container.

The tutorial describes the creation of Java servlets. The usage of JavaServer Pages is widely seen as outdated, other frameworks are used for the creation of web user interfaces.

**2. Tomcat Installation**

Eclipse allows downloading and starting a local Tomcat installation within the Eclipse IDE. Their is no need for an additional setup.

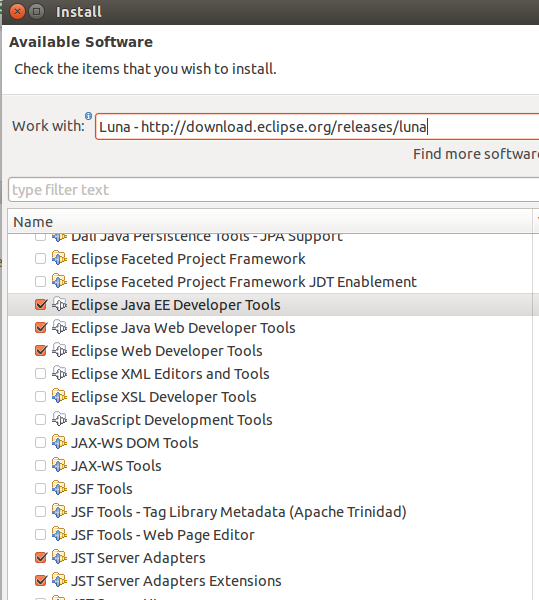
If you have a local running Tomcat installation on your computer, stop Tomcat. Eclipse WTP needs to start Tomcat itself for its deployments.

**3. Installation of WTP**

Your version of the Eclipse IDE might not have the required web developer tools installed.

Use the [**Eclipse Update Manager**](http://www.vogella.com/tutorials/Eclipse/article.html#updatemanager)to install the following features from the "Web, XML, Java EE Development and OSGi Enterprise Development" category.

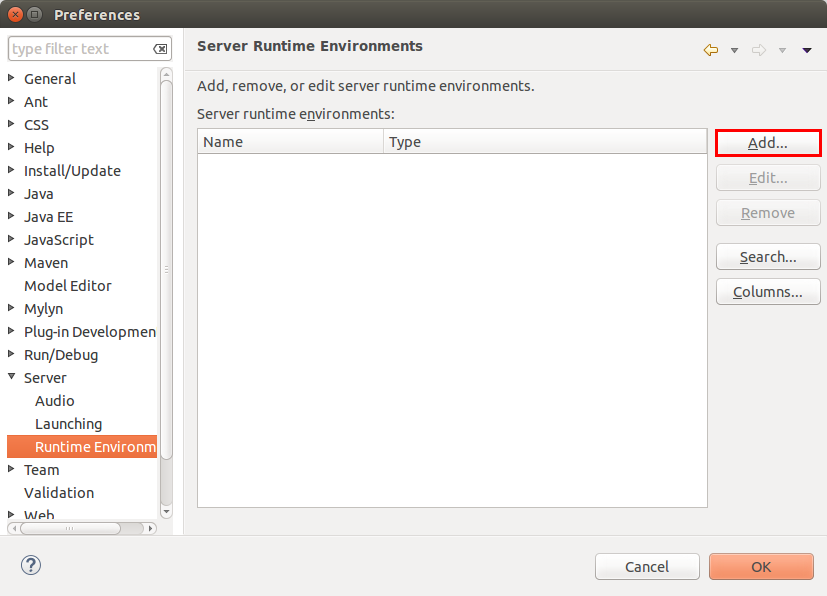
* Eclipse Java EE Developer Tools
* Eclipse Java Web Developer Tools
* Eclipse Web Developer Tools
* JST Server Adapters
* JST Server Adapters Extensions



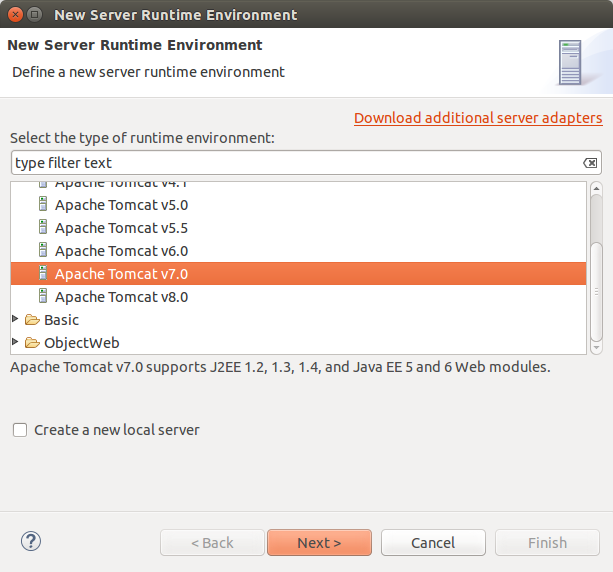
**4. WTP Configuration**

**4.1. Setting up runtime environments**

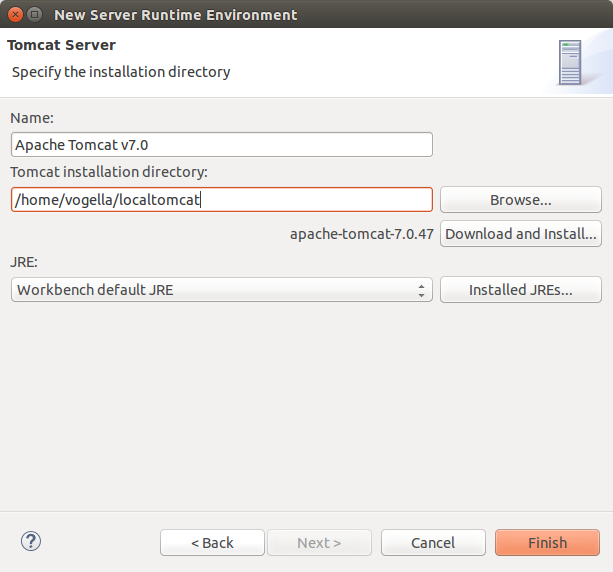
To configure Eclipse WTP, select from the menu *Window* → *Preferences* → *Server* → *Runtime Environments*. Press the *Add* button.



Select your version of Tomcat and select the *Create a new local server* flag.



You can instruct Eclipse to download Tomcat into a local directory.

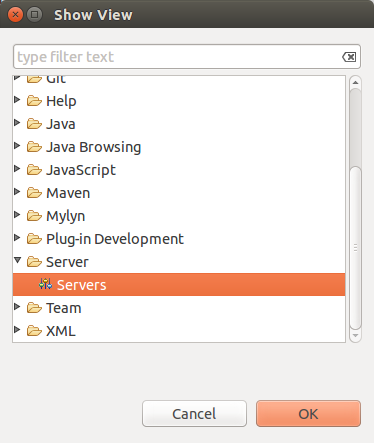


Press the *Finish* and afterwards OK.

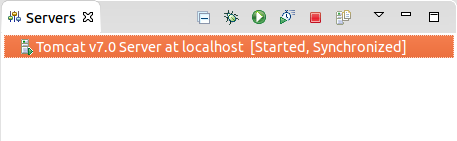
You are now ready to use Tomcat together with the Eclipse IDE.

**4.2. Server**

Create a local Tomcat server. During development, you will create your server. You can manage your server via the *Servers* view. To open this view select *Window* → *Show View* → *Other...* → *Server* → *Servers...*.



You can stop and start the Tomcat server via the *Servers* view.



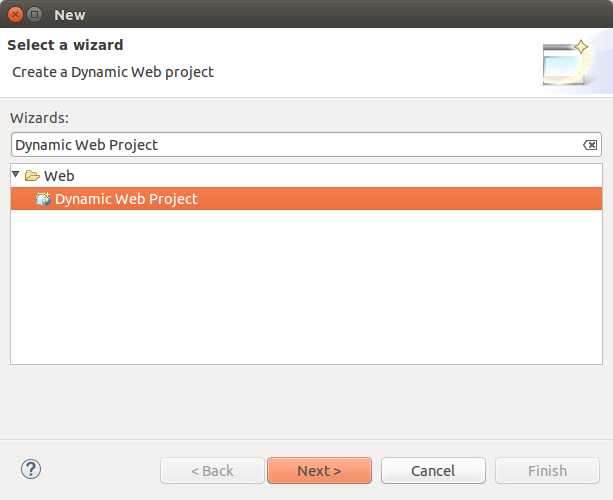
**5. Exercise: Create a servlet**

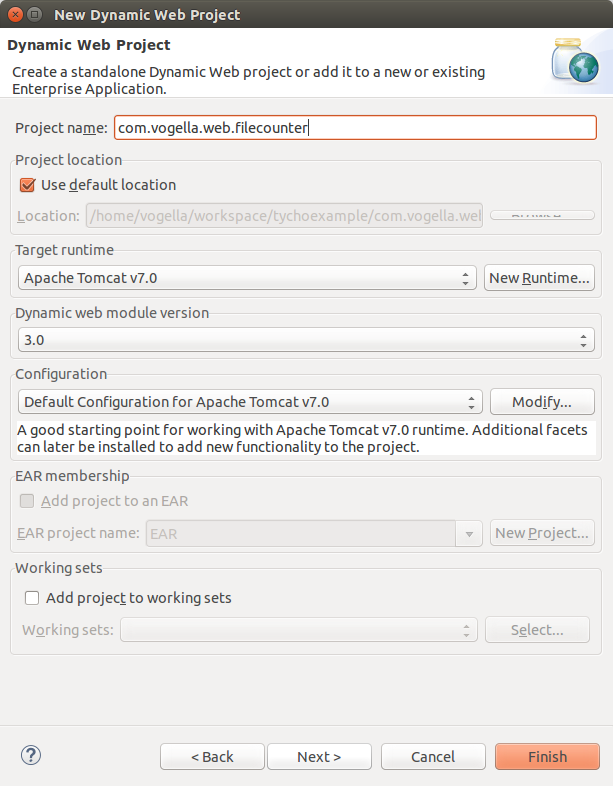
**5.1. Target**

In this exercise you create a servlet which works as a webpage counter. This servlet keeps track of the number of visitors of a webpage. The servlet persists the number of visitors in a text file.

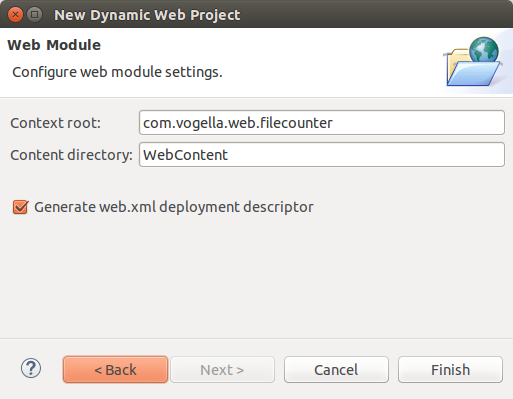
**5.2. Create web project**

Create a new *Dynamic Web Project* called *com.vogella.web.filecounter* by selecting *File* → *New* → *Other...* → *Web* → *Dynamic Web Project*.





Press twice the *Next* button and select the *Generate web.xml deployment descriptor* on the last page.



Afterwards press the *Finish* button.

If Eclipse asks you, to switch to the *Java EE Perspective* answer yes.

A new project has been created with the standard structure of a Java web application. The *WEB-INF/lib* directory holds all the JAR files that the Java web application requires.

**5.3. Creating Data Access Object**

Create a new package called *com.vogella.web.filecounter.dao*.

Create the following new Java class to read and write the counter value to and from the file.

**package** com.vogella.web.filecounter.dao;

**import** java.io.BufferedReader;

**import** java.io.File;

**import** java.io.FileReader;

**import** java.io.FileWriter;

**import** java.io.IOException;

**import** java.io.PrintWriter;

**public** **class** FileDao {

**public** **int** getCount() {

**int** count = 0;

*// Load the file with the counter*

FileReader fileReader = null;

BufferedReader bufferedReader = null;

PrintWriter writer = null;

**try** {

File f = **new** File("FileCounter.initial");

**if** (!f.exists()) {

f.createNewFile();

writer = **new** PrintWriter(**new** FileWriter(f));

writer.println(0);

}

**if** (writer != null) {

writer.close();

}

fileReader = **new** FileReader(f);

bufferedReader = **new** BufferedReader(fileReader);

String initial = bufferedReader.readLine();

count = Integer.parseInt(initial);

} **catch** (Exception ex) {

**if** (writer != null) {

writer.close();

}

}

**if** (bufferedReader != null) {

**try** {

bufferedReader.close();

} **catch** (IOException e) {

e.printStackTrace();

}

}

**return** count;

}

**public** **void** save(**int** count) **throws** Exception {

FileWriter fileWriter = null;

PrintWriter printWriter = null;

fileWriter = **new** FileWriter("FileCounter.initial");

printWriter = **new** PrintWriter(fileWriter);

printWriter.println(count);

*// make sure to close the file*

**if** (printWriter != null) {

printWriter.close();

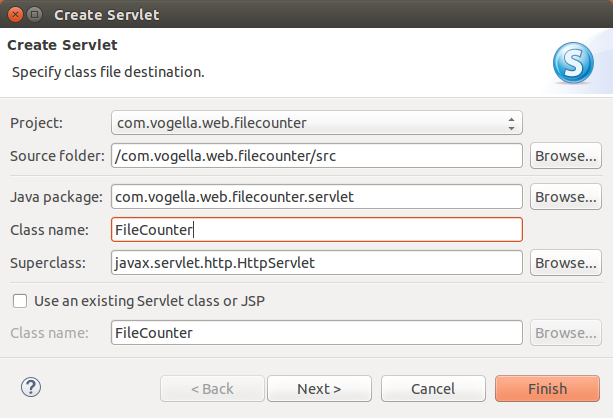
}

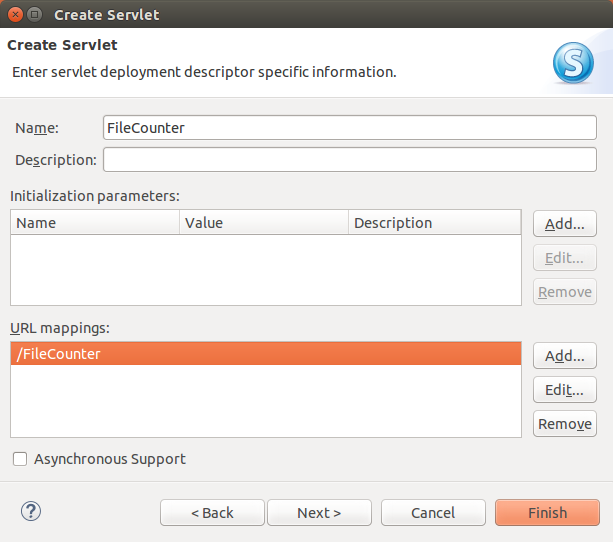
}

}

**5.4. Creating the Servlet**

Create a servlet. Right-click on your project and select *New* → *Servlet*. Enter the following data.





Press finish.

You could also create a servlet without the wizard. The wizard creates a Java class which extends thejavax.servlet.http.HttpServlet and adds the servlet settings to the *web.xml* file.

Enter the following code.

**package** com.vogella.web.filecounter.servlet;

**import** java.io.IOException;

**import** java.io.PrintWriter;

**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.servlet.http.HttpSession;

**import** com.vogella.web.filecounter.dao.FileDao;

/\*\*

\* Servlet implementation class FileCounter

\*/

*@WebServlet("/FileCounter")*

**public** **class** FileCounter **extends** HttpServlet {

**private** **static** **final** **long** serialVersionUID = 1L;

**int** count;

**private** FileDao dao;

*@Override*

**protected** **void** doGet(HttpServletRequest request,

HttpServletResponse response) **throws** ServletException, IOException {

*// Set a cookie for the user, so that the counter does not increate*

*// every time the user press refresh*

HttpSession session = request.getSession(true);

*// Set the session valid for 5 secs*

session.setMaxInactiveInterval(5);

response.setContentType("text/plain");

PrintWriter out = response.getWriter();

**if** (session.isNew()) {

count++;

}

out.println("This site has been accessed " + count + " times.");

}

*@Override*

**public** **void** init() **throws** ServletException {

dao = **new** FileDao();

**try** {

count = dao.getCount();

} **catch** (Exception e) {

getServletContext().log("An exception occurred in FileCounter", e);

**throw** **new** ServletException("An exception occurred in FileCounter"

+ e.getMessage());

}

}

**public** **void** destroy() {

**super**.destroy();

**try** {

dao.save(count);

} **catch** (Exception e) {

e.printStackTrace();

}

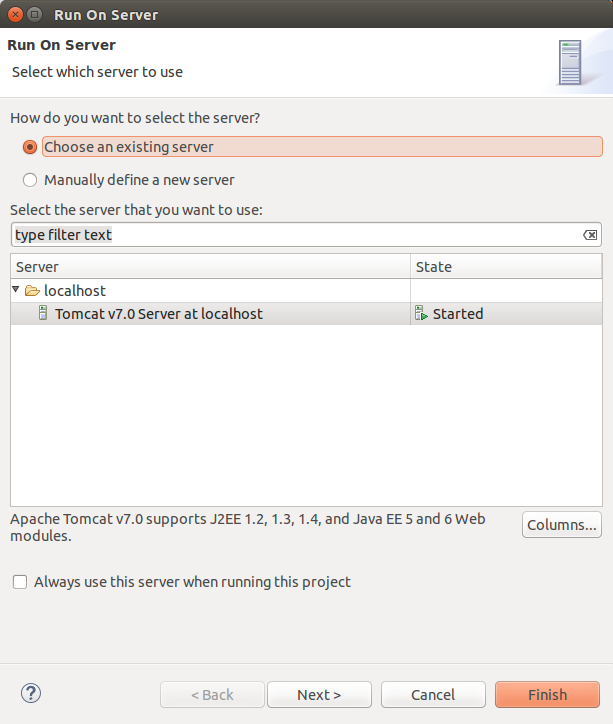
}

}

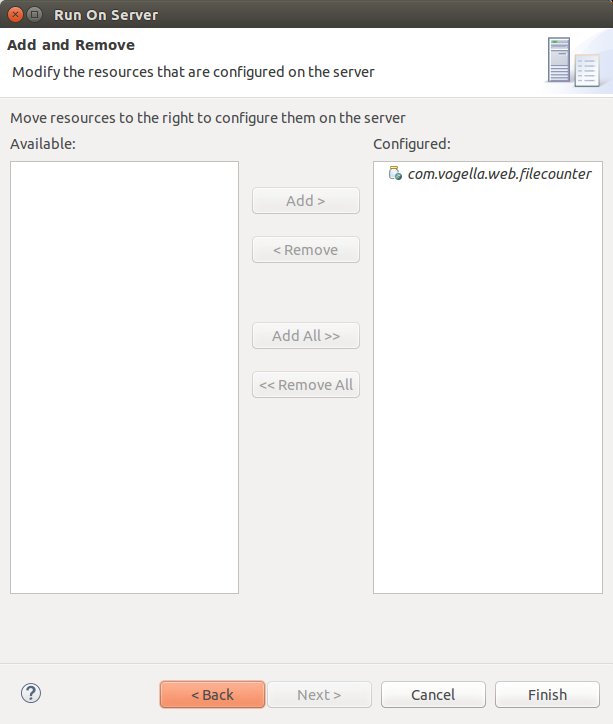
This code will read the counter from a file on the server and return plain text to the browser. The servlet will increase the counter if the user was inactive for 5 seconds.

**5.5. Deployment of the servlet**

You application must be deployed to the server and started there. Select your servlet or your project, right-click on it and select *Run As* → *Run on Server* from the context menu and select your existing server.



Ensure your servlet is selected to run on the server.



**Note**

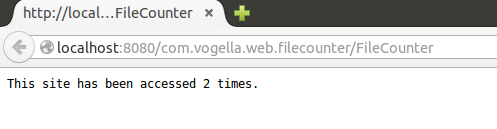
In case you get the error "The Tomcat server configuration at \Servers\Tomcat v7.0 Server at localhost-config is missing. Check the server for errors.", simply delete the existing server configuration and create a new one.

Press the *Finish* button.

**5.6. Validate the deployment**

If the deployment was successfully you should be able to access your servlet via the following URL: http://localhost:8080/com.vogella.web.filecounter/FileCounter

The servlet you return the number of times it was accessed. If you wait 5 seconds and refresh the browser, the number should increase.

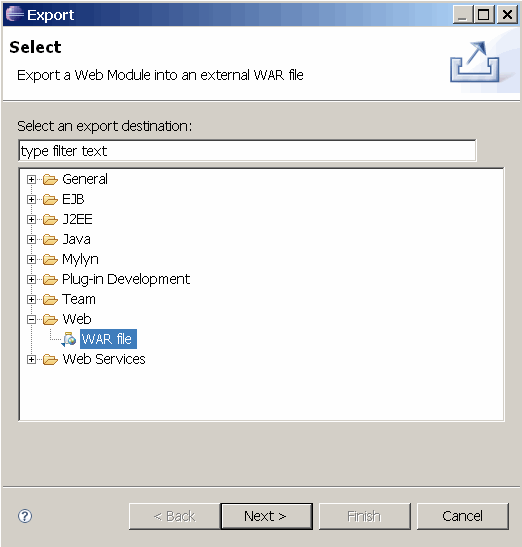


Congratulations. You created your first working servlet with Eclipse WTP!

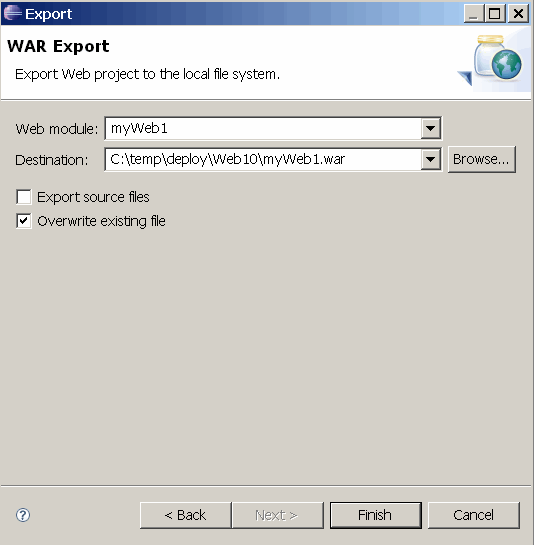
**6. Web Archive - How to create a war file from Eclipse**

´The following describes how to create a Web Archive (WAR) from Eclipse.

Right-click on the project and select *Export*.



Specify the target directory and press *Finish*.



You can now import the WAR file to your production Tomcat system and test the web application.