FAQ: HELP! My Spring MVC Controller is not working. What to do?

JUST A HEADS UP ...

IN THE VIDEOS YOUR SPRING MVC CONTROLLER CODE MAY NOT WORK

THIS IS AN ECLIPSE CACHING BUG

IF YOUR CONTROLLER DOESN'T WORK THEN TRY THE STEPS BELOW

Question: HELP! My Spring MVC Controller is not working. What to do?

Answer:

Debug Tip #0: Make sure you are accessing the correct URL

You need to access the correct URL, localhost:8080/spring-mvc-demo/

Note: Do not attempt to run the JSP files directly. This will not work due to MVC.

Debug Tip #1: Make Sure your Package Names is consistent

The Spring configuration file used in the videos expects for your controller to be in the package:

package com.luv2code.springdemo.mvc;

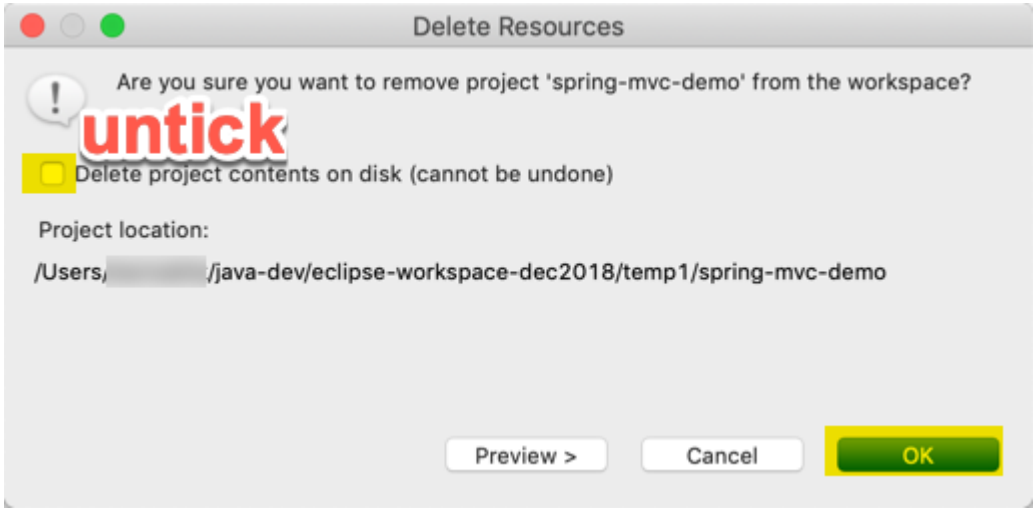
Make sure your controller is this package.

Debug Tip #2: Reimport Project into Eclipse

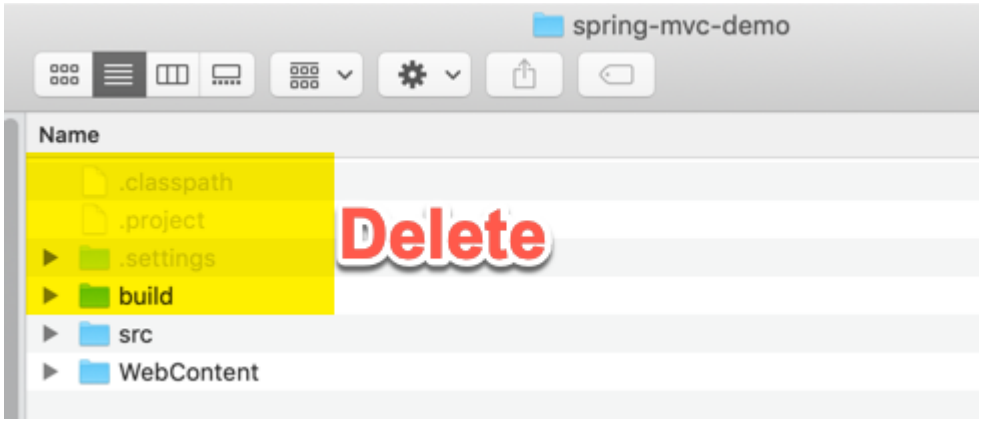
Sometimes the Eclipse project settings can become corrupted. One possible solution is to remove the Eclipse project and reimport into Eclipse.

Here are the steps.

1. Remove the project from Eclipse.



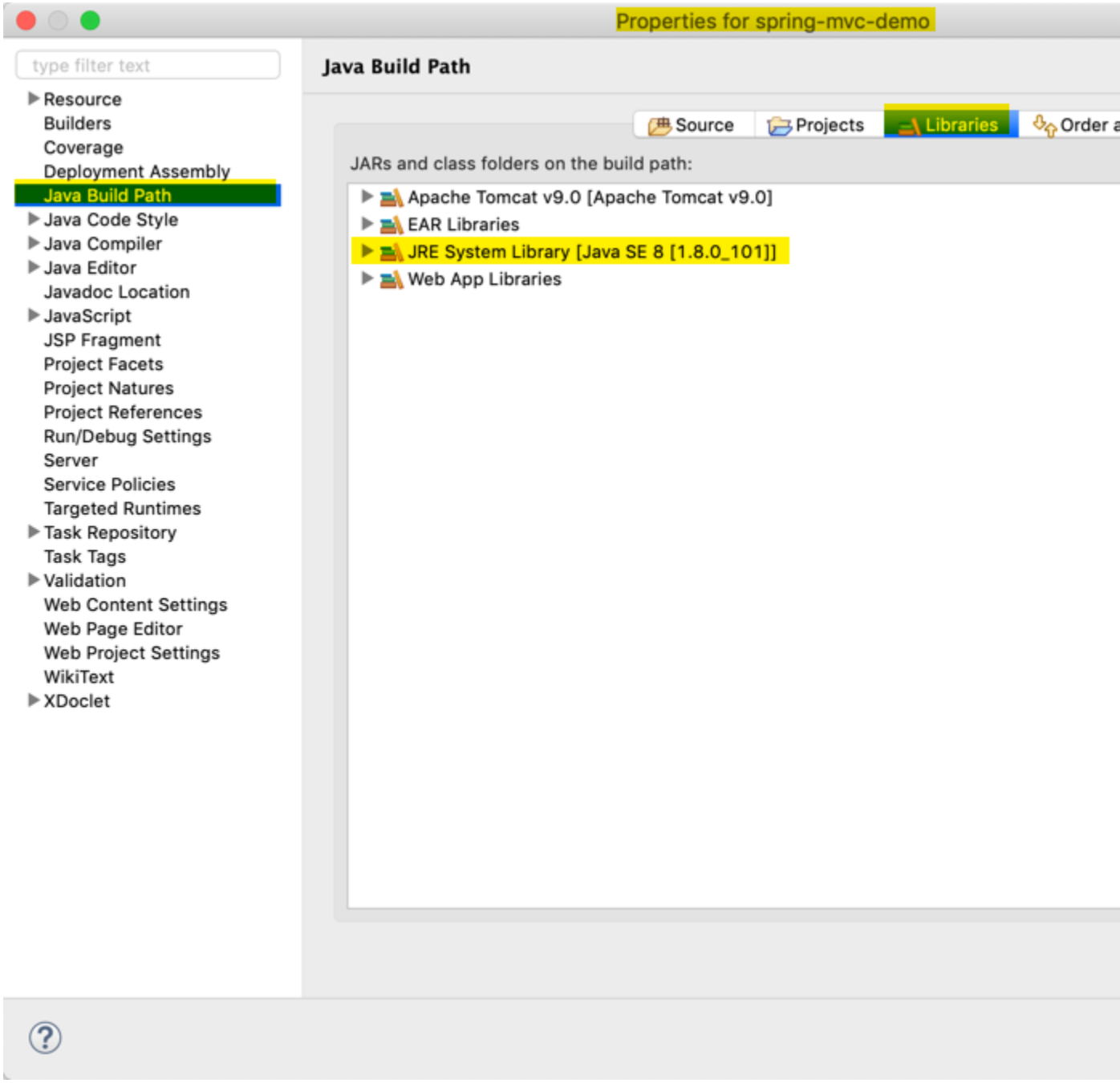
2. Open the project in file system and remove the files/ folders except src and WebContent.

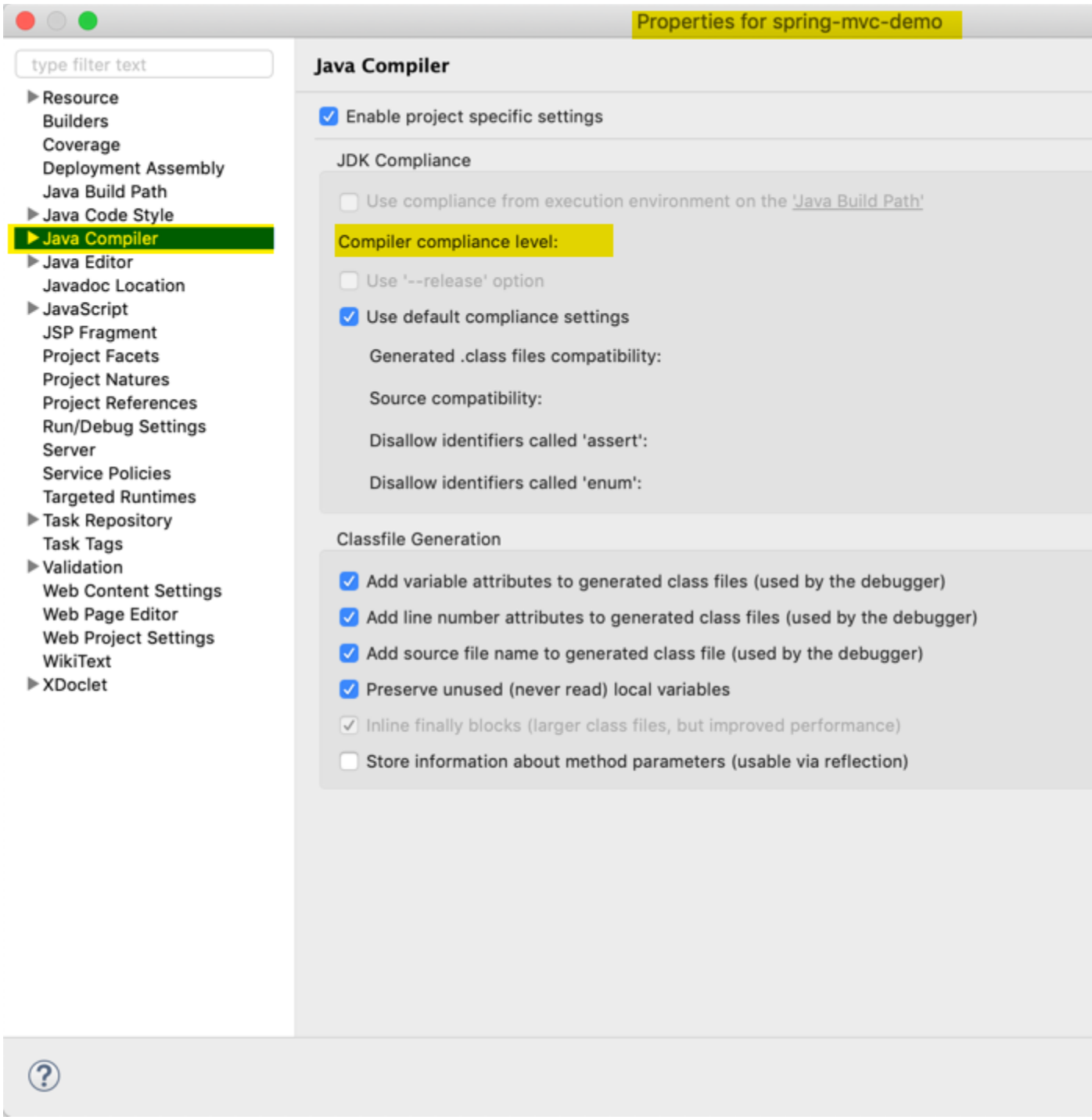


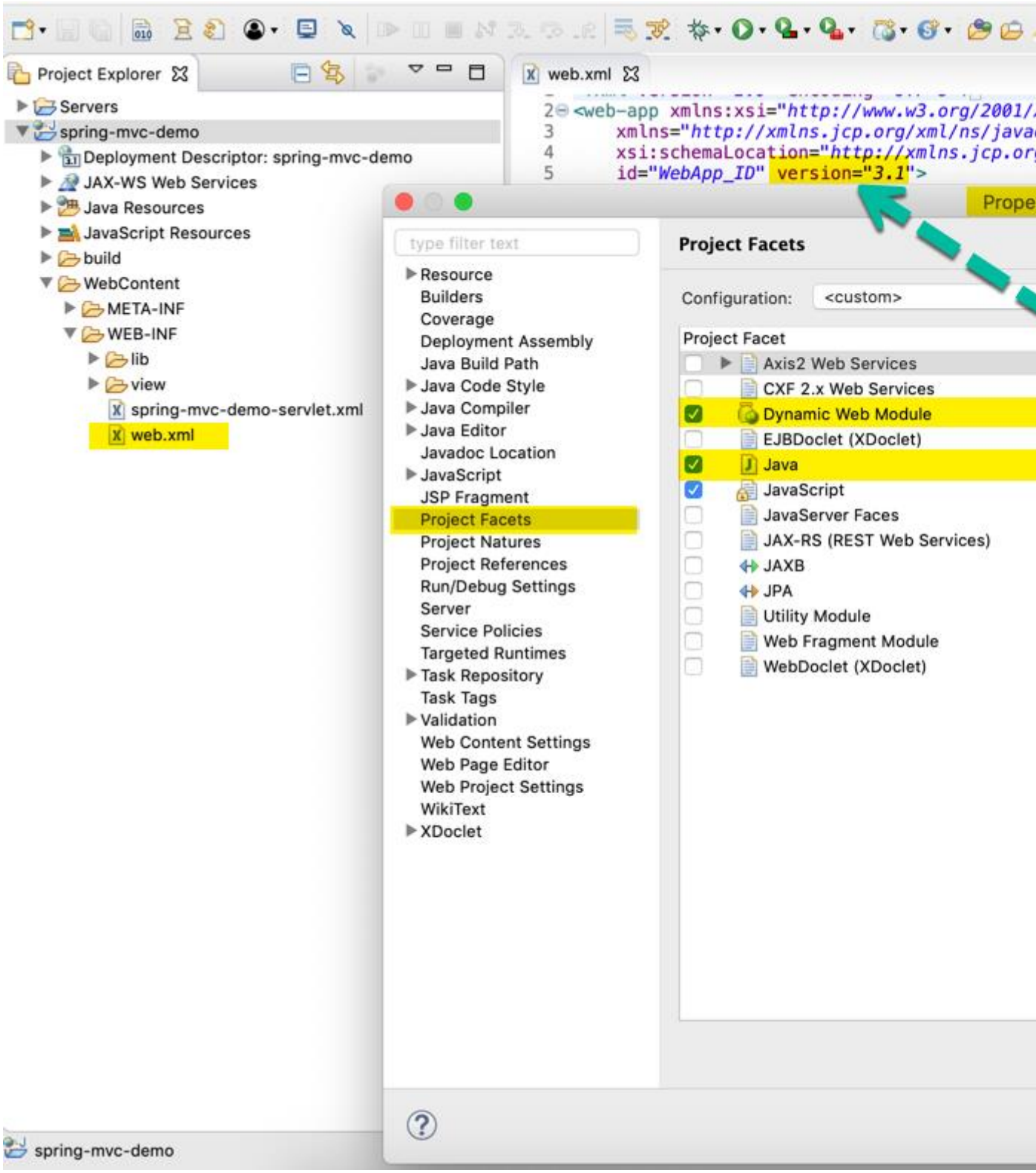
3. Open the project in Eclipse again using **File > Import ... > General > Existing Projects into Workspace**

4. Check the project properties(Right click on the project and click on Properties),

5. Check the Java Build Path, Java Compiler and Project Facets and make sure that the Java version is same in all the windows.




Properties for spring-mvc-demo
type filter text
Resource
Builders
Coverage
Deployment Assembly
Java Build Path
Java Code Style
Java Compiler
Java Editor
Javadoc Location
JavaScript
JSP Fragment
Project Facets
Project Natures
Project References
Run/Debug Settings
Server
Service Policies
Targeted Runtimes
Task Repository
Task Tags
Validation
Web Content Settings
Web Page Editor
Web Project Settings
WikiText
XDoclet
Java Compiler
Enable project specific settings
JDK Compliance
Use compliance from execution environment on the 'Java Build Path'
Compiler compliance level:
Use '--release' option
Use default compliance settings
Generated .class files compatibility:
Source compatibility:
Disallow identifiers called 'assert':
Disallow identifiers called 'enum':
Classfile Generation
Add variable attributes to generated class files (used by the debugger)
Add line number attributes to generated class files (used by the debugger)
Add source file name to generated class file (used by the debugger)
Preserve unused (never read) local variables
Inline finally blocks (larger class files, but improved performance)
Store information about method parameters (usable via reflection)



5. Build the project and try again.

Debug Tip #3: Clear Tomcat Cache

This is normally a caching issue with the cache.

Here are some steps to clear the Eclipse cache and Tomcat cache.

1. In the Server's tab in Eclipse, Stop the Tomcat server

2. Right-click the server and select "Clean..."

3. Right-click the server again and select "Clean Tomcat Work Directory..."

---

4. In the Eclipse, select the top-level menu option, Project > Clean ...

5. Be sure your project is selected and click Ok

6. Restart Eclipse

Retest your application. If you continue to have problems try Debug Tip #2

Debug Tip #4: Import the Project in a new workspace

1. Copy your project to a new directory on your computer like, c:\foobar

2. In Eclipse, open a new workspace: Create a new workspace in Eclipse: File > Switch Workspace > Other ... > give any name

3. In Eclipse, import the project

3a. Use File > Import > General > Existing Projects into Workspace

3b. Browse to directory: c:\foobar

4. In your new workspace, add a reference to the Tomcat server

5. Test your app in the new workspace

If you still have problems, then post your code to the classroom discussion forum.

Be sure to include the following files:
- spring-mvc-demo-servlet.xml
- your controller .java
- your view page .jsp

Resources for this lecture

- solution-code-spring-mvc-create-home-controller-and-view.zip