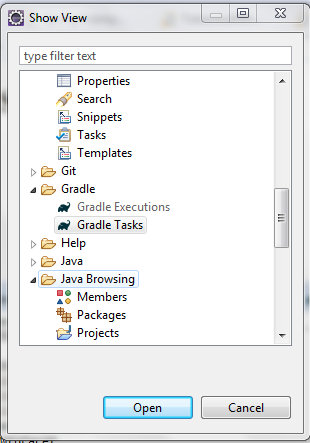
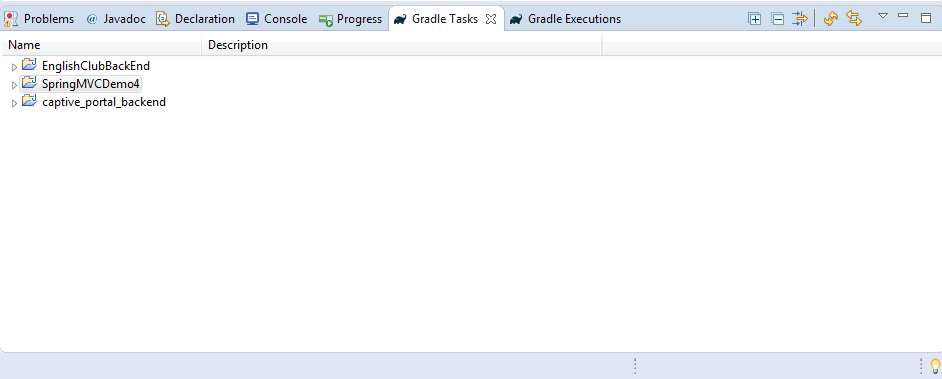
Deploying Spring MVC Application on Apache Tomcat Server

**\* Note: This technique is only used for Tomcat 7 or newer.**

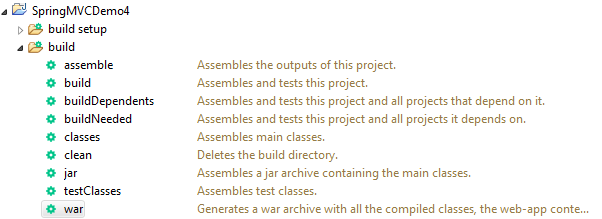
1. From Eclipse toolbar, select Window -> Show Views -> Other. Search for Gradle Tasks.



Open that Show View, and the Gradle Task tab will appear in the bottom panel of Eclipse like this:

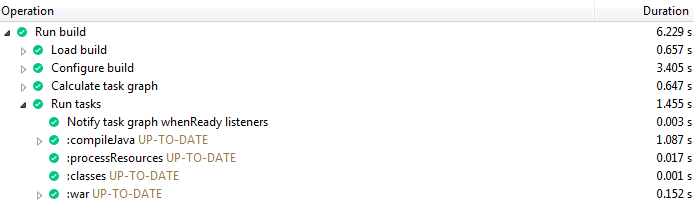


1. In this picture above, you will see your project in there (this is SpringMVCDemo4 in this example). Click on this project and choose build -> war.

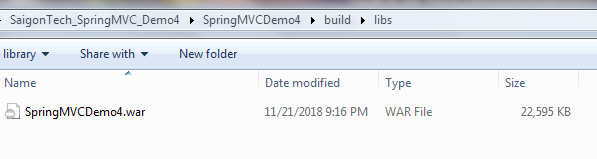


This will pack your project into a .war file. This is the type of file that will be readed by Apache Tomcat Web Service.

Wait until this task done successfully like this:

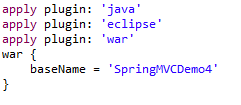


This .war file will be created in <Your Project Dir>/build/libs/

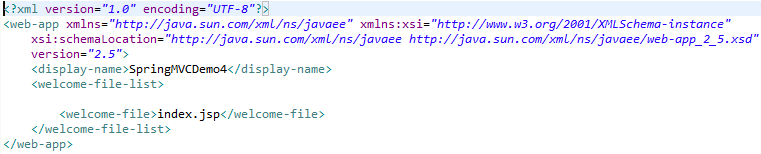


Note: we must not rename this file normally, because it has been set up with your project to be compatible with Tomcat. On the other hand, we can rename this .war by editting the attribute ***war-baseName*** in the project's ***build.gradle*** file and attribute ***<display-name>*** in ***web.xml***:

**build.gradle**

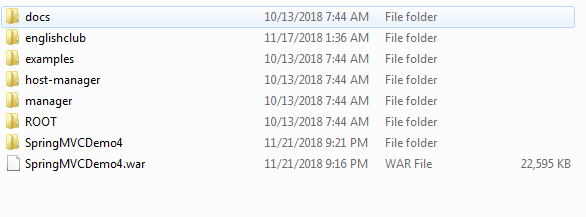


**web.xml**



After some research, I think that we should consider your project name from the beginning. Therefore, it will keep you safe from deploying errors.

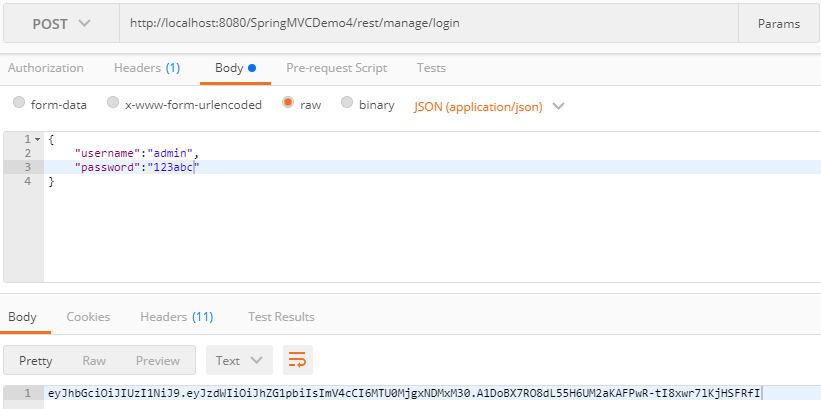
1. Copy or Cut this .war file to <Tomcat-dir>/webapps/ folder and re-run your Tomcat. This will generate a new folder, that has the name same with your .war file



1. Now we can access our project via Tomcat Web Server by this URL:

*http://localhost:8080/<War-file-name>/<Your-Controller-you-declared-in-your-project>*

This is an example of login controller:



In short, this is the way that many companies host their Spring MVC Application on their Server.

End.