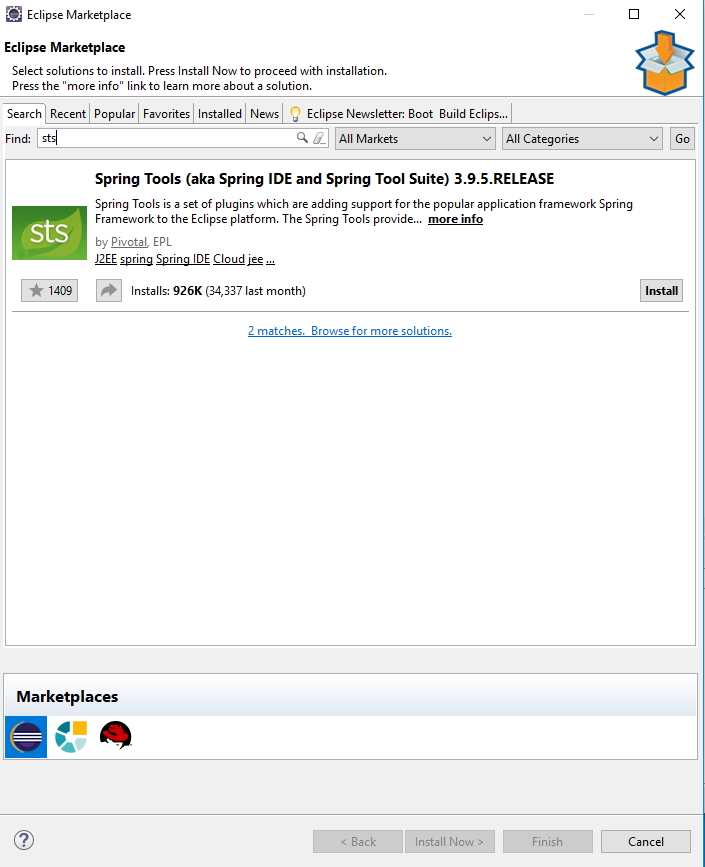
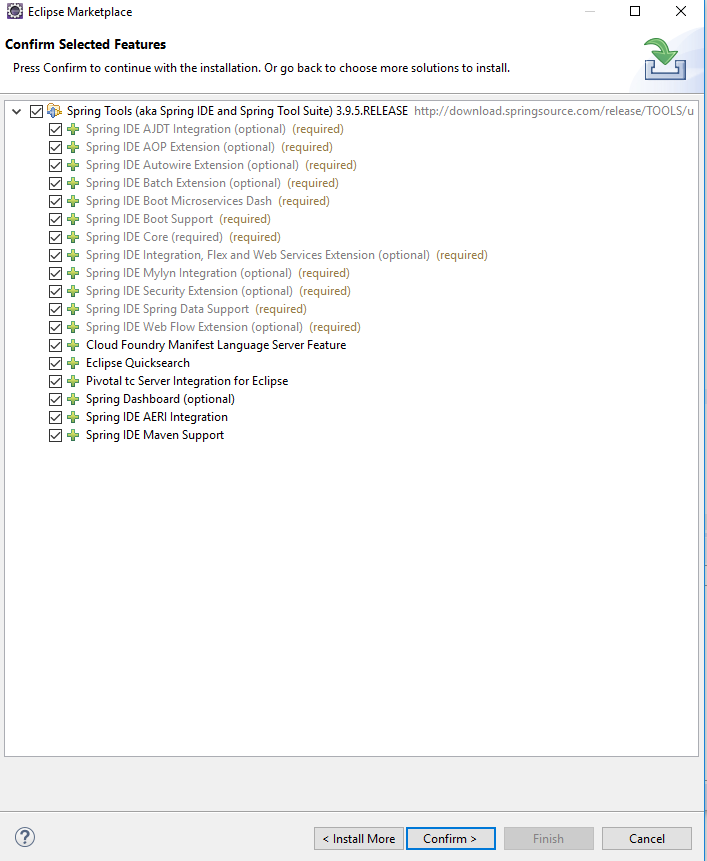
Running the Stubs Project in Eclipse

## Configuring and Running the Stubs and Proxy Project

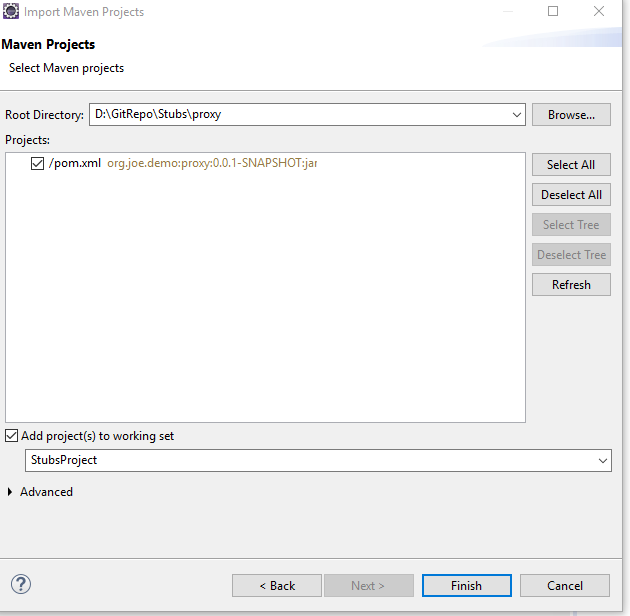
1. Run a git clone on the Stubs url in a new directory e.g. D:/GitRepo
2. Checkout a new local branch e.g. Leland/test
3. Open Eclipse (version should be at least Eclipse Oxygen) and install the STS plugin in eclipse marketplace:



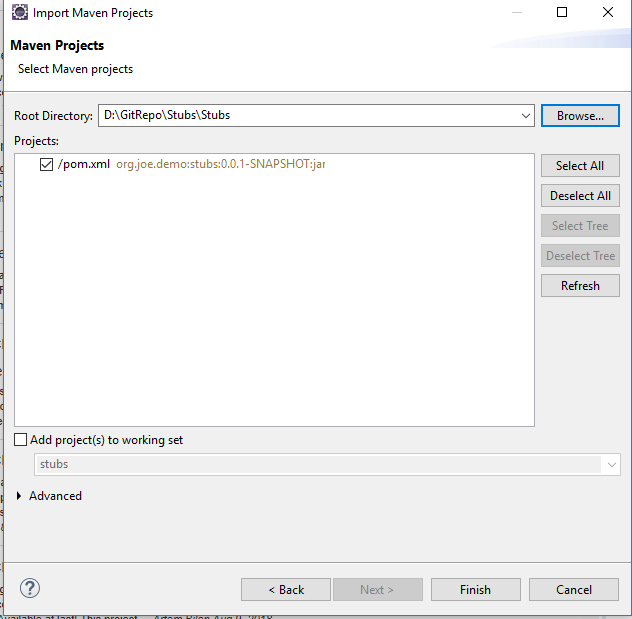
1. Leave all options as default and click “Confirm”:



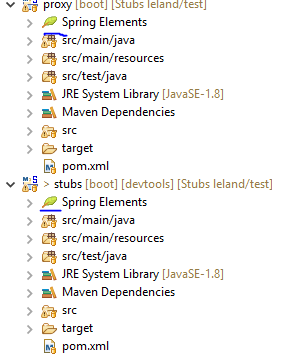
1. Accept the terms and click “finish”. In the process of installing, agree to install unsigned content. Restart eclipse when finished.
2. Use the import wizard to import an existing maven project into the eclipse workspace. Select the “Proxy” project and click “Finish”:



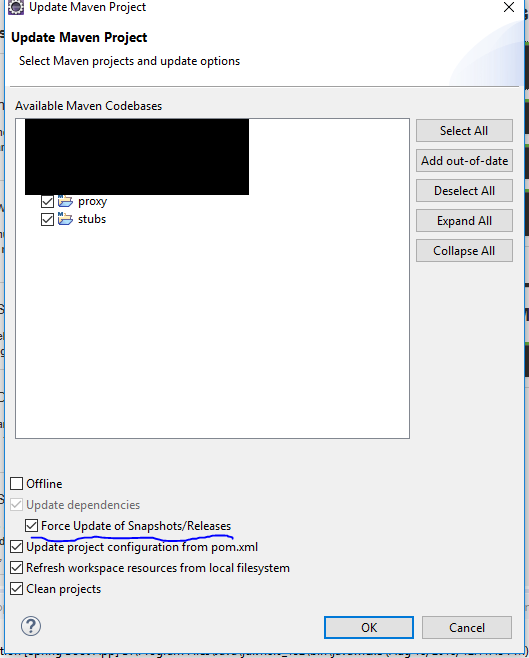
1. Repeat step 5 for the “stubs” project and click “Finish”:



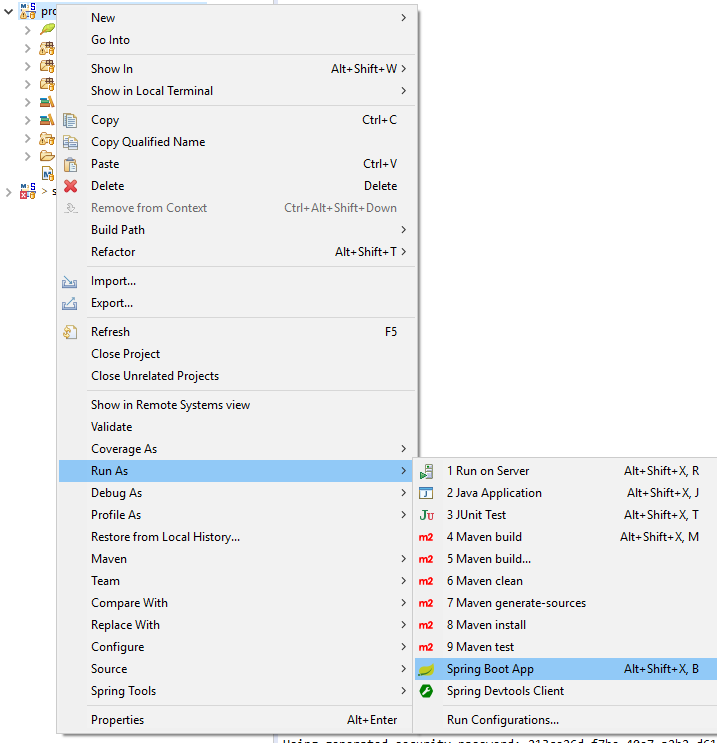
1. After the workspace is finished building, expand both the proxy and stubs projects. If configured correctly, Eclipse will recognize them as spring boot projects and add the “Spring Elements” dropdown:



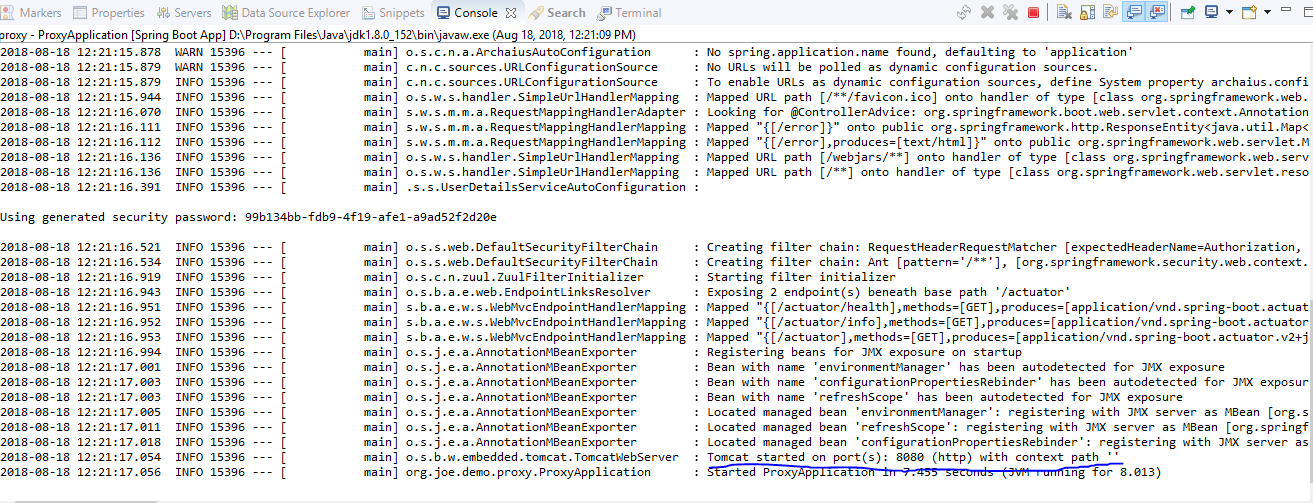
1. Run a maven update on both projects. When prompted, select “Force Update of Snapshots/Releases”.



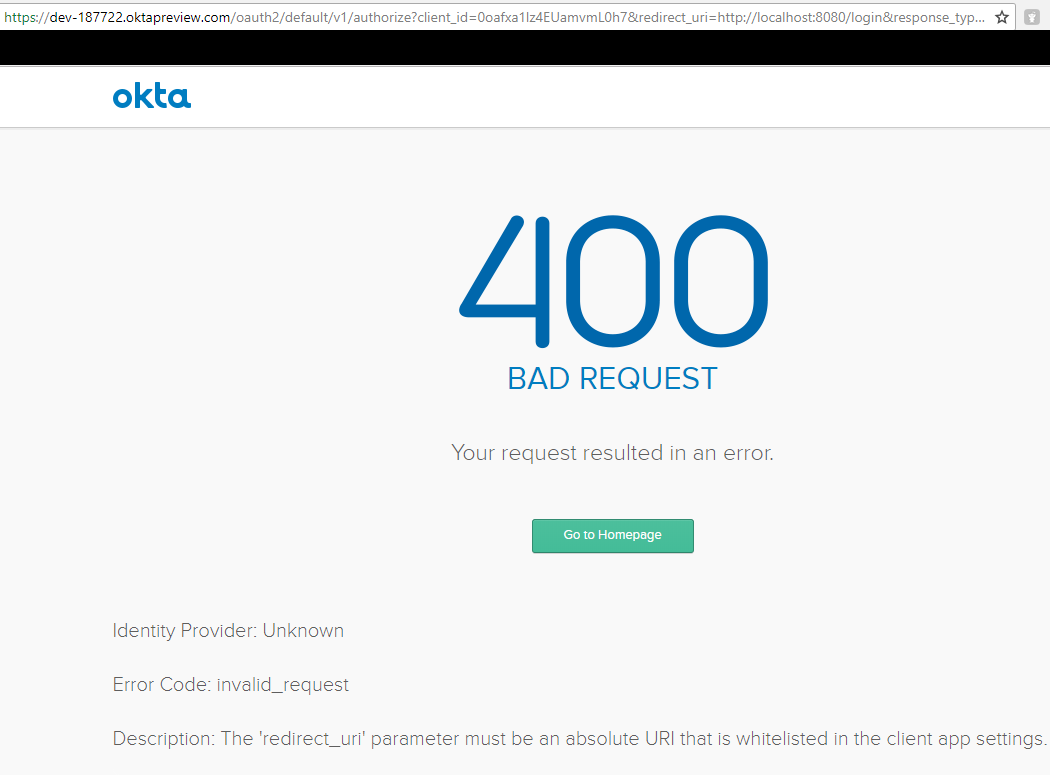
1. Use Eclipse to run “clean maven install -U” on the proxy project and the stubs project in that order.
2. When finished installing, the projects are ready to be booted using Spring. Right click the “proxy” project and select “Run as > Spring Boot App”



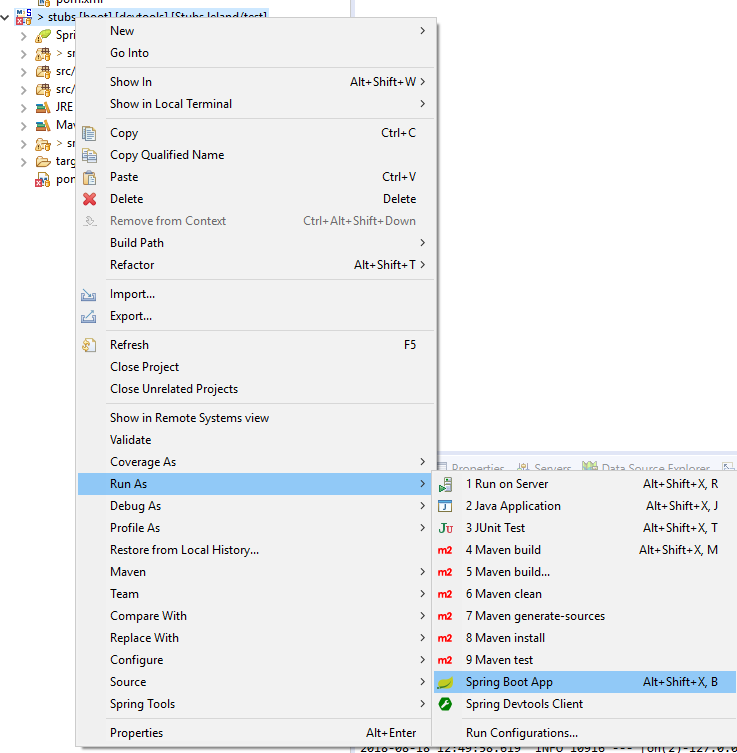
1. When the application is finished booting it will display the console message “Tomcat started on port(s): 8080 (http) with context path '',” which means the application has been successfully launched and can be accessed at “http:localhost/8080”.



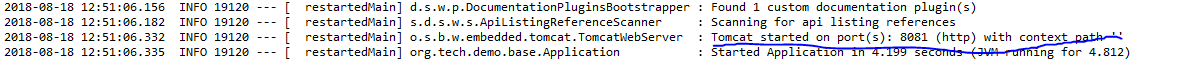
1. Open your web browser and navigate to “http:localhost/8080”. If the project is successfully configured, you will be redirected to a 400 error:



1. Repeat step 11 above to boot the “stubs” project:



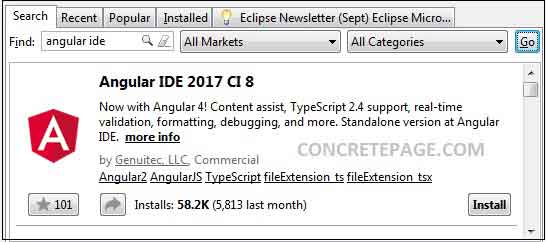
1. When the application is finished booting it will display the console message “Tomcat started on port(s): 8081 (http) with context path '',” which means the application has been successfully launched and can be accessed at “http:localhost/8081”.



1. Open your web browser and navigate to “http:localhost/8081”. If the project is successfully configured, a json document will be displayed:

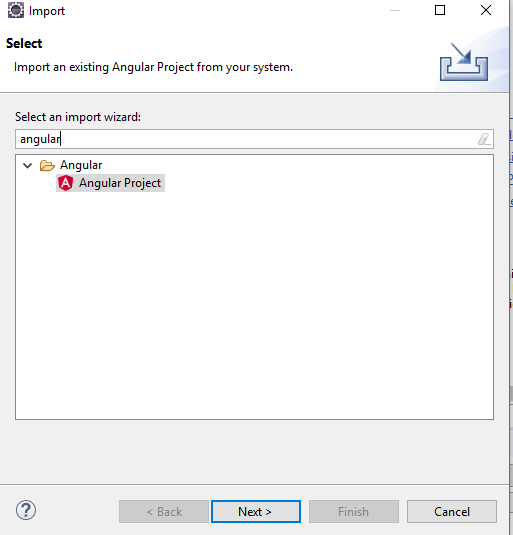


## Configuring and Running the Client Project

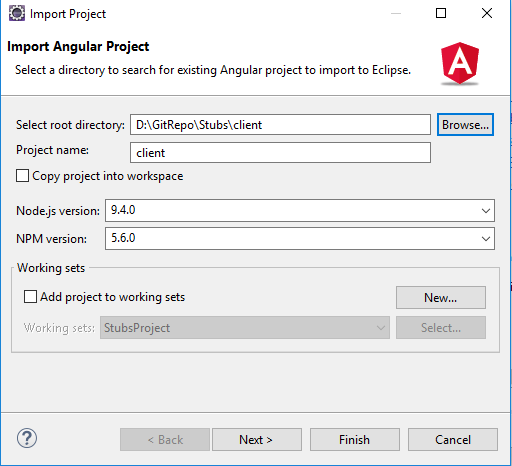
1. From inside Eclipse, open the eclipse marketplace and install the Angular IDE plugin:
2. 

[Source](https://www.concretepage.com/angular-2/angular-2-4-angular-ide-eclipse-hello-world-example#Install)

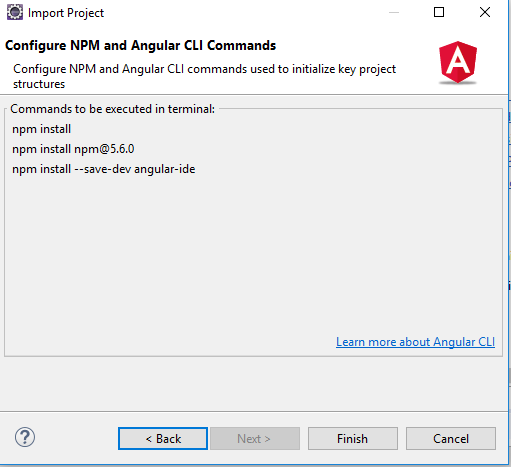
1. Click through the screens.
2. Use Eclipse’s import wizard to import the client project as an angular project:



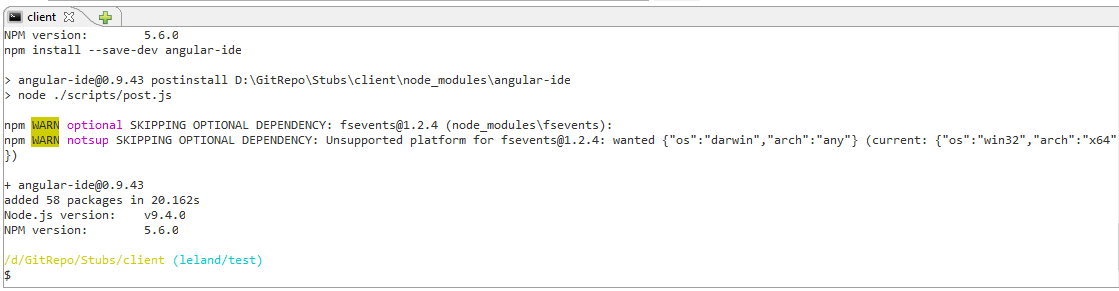
1. On the next Screen navigate to the client project and leave all options as the default and click next.

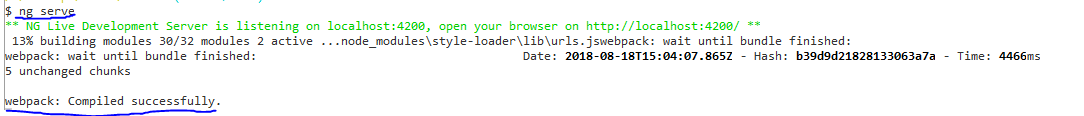


1. At this point, the Angular IDE will prompt the user that it will install any needed project structures. Click “Finish”:



1. After hitting “Finish” eclipse will prompt the user to open the Angular perspective. For purposes of this tutorial click “yes”. The installation commands will run in Eclipse’s Terminal+ view. Once finished, Terminal+ will prompt the user with a shell in the cwd:



1. Run “ng serve” in Terminal+ to start the Angular app. When user receives the message “webpack compiled successfully”: 

navigate to <http://localhost:4200/home> in your internet browser to access the app:

