

Appendix: Instructions for IntelliJ IDEA Users

This section of the lab documentation provides a set of instructions on how to use IntelliJ IDEA as an alternative to STS during the course.

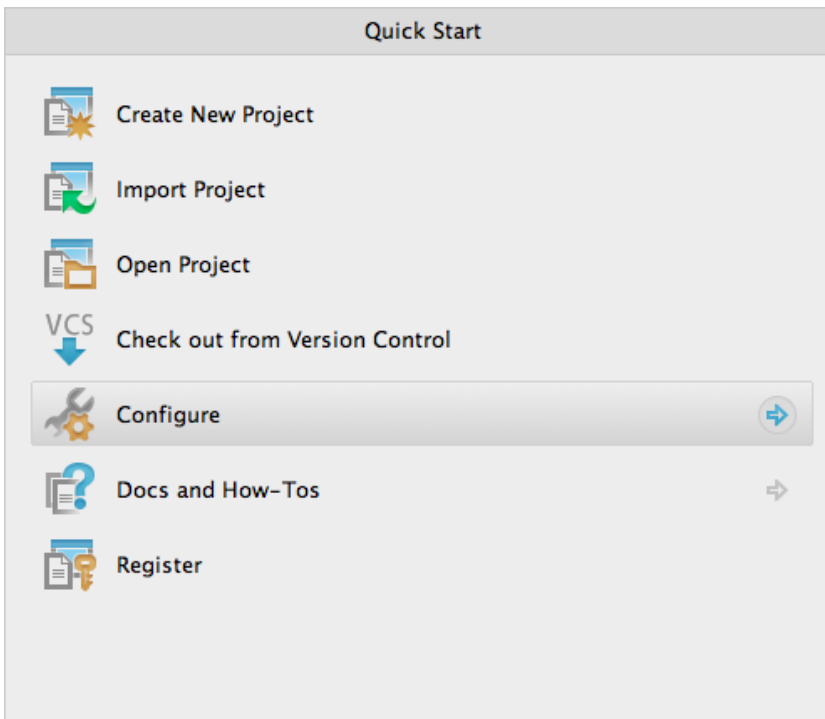
Configuring the IDE

Configure a JDK

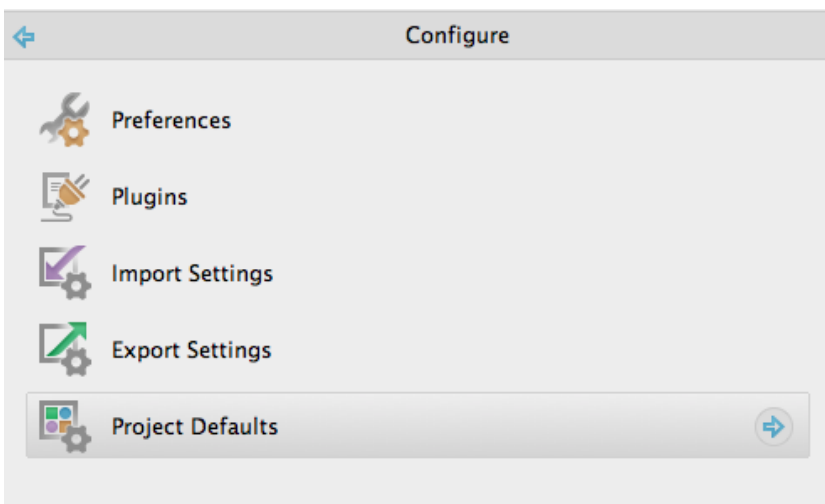
The first thing you have to do after installing IntelliJ IDEA is configure a JDK.

To see the list of preconfigured JDKs click the *Configure* button in the *Quick Start* panel of the *Welcome screen*.

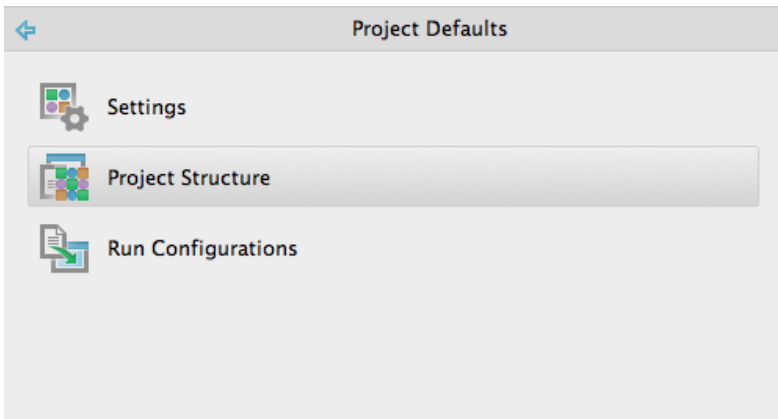
The window contents slide across.



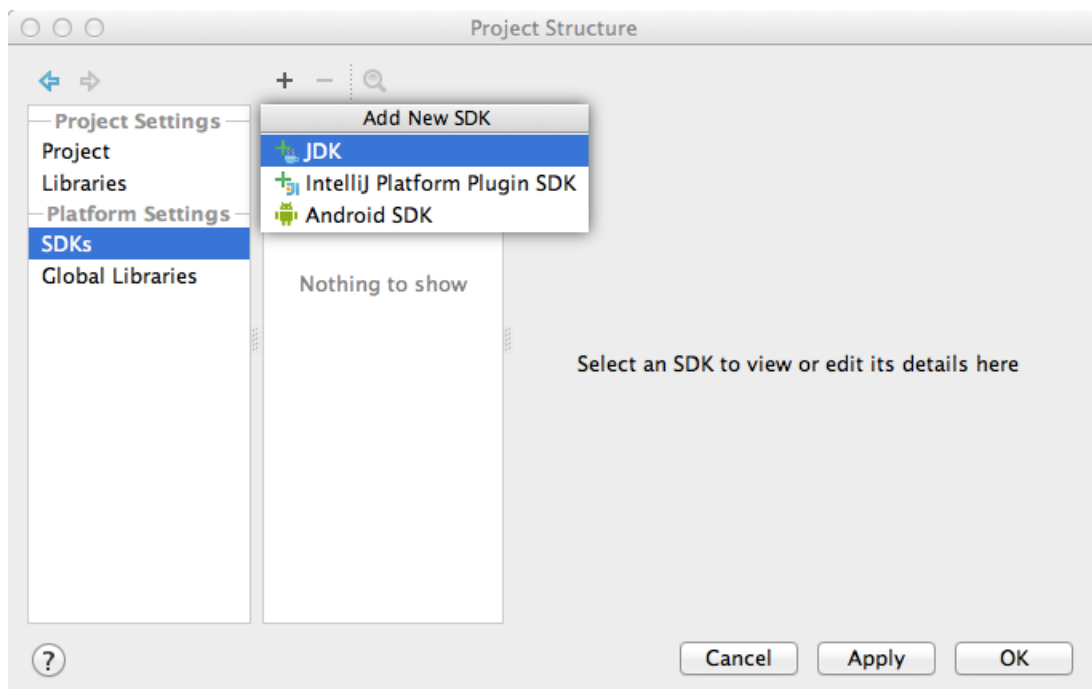
Now choose *Project Defaults*



And finally *Project Structure*



In the *Project Structure* dialog switch to the *SDKs* section. Here you can add a JDK by clicking the plus button:



Make sure you have configured at least one JDK before you import the project.

Specify Maven local repository

Before you import the project, be sure to specify the Maven repository found within the courseware installation folder or directory. To do that, click the *Configure* button on the *Welcome screen*, and then choose Settings (or *Preferences* for MacOS).

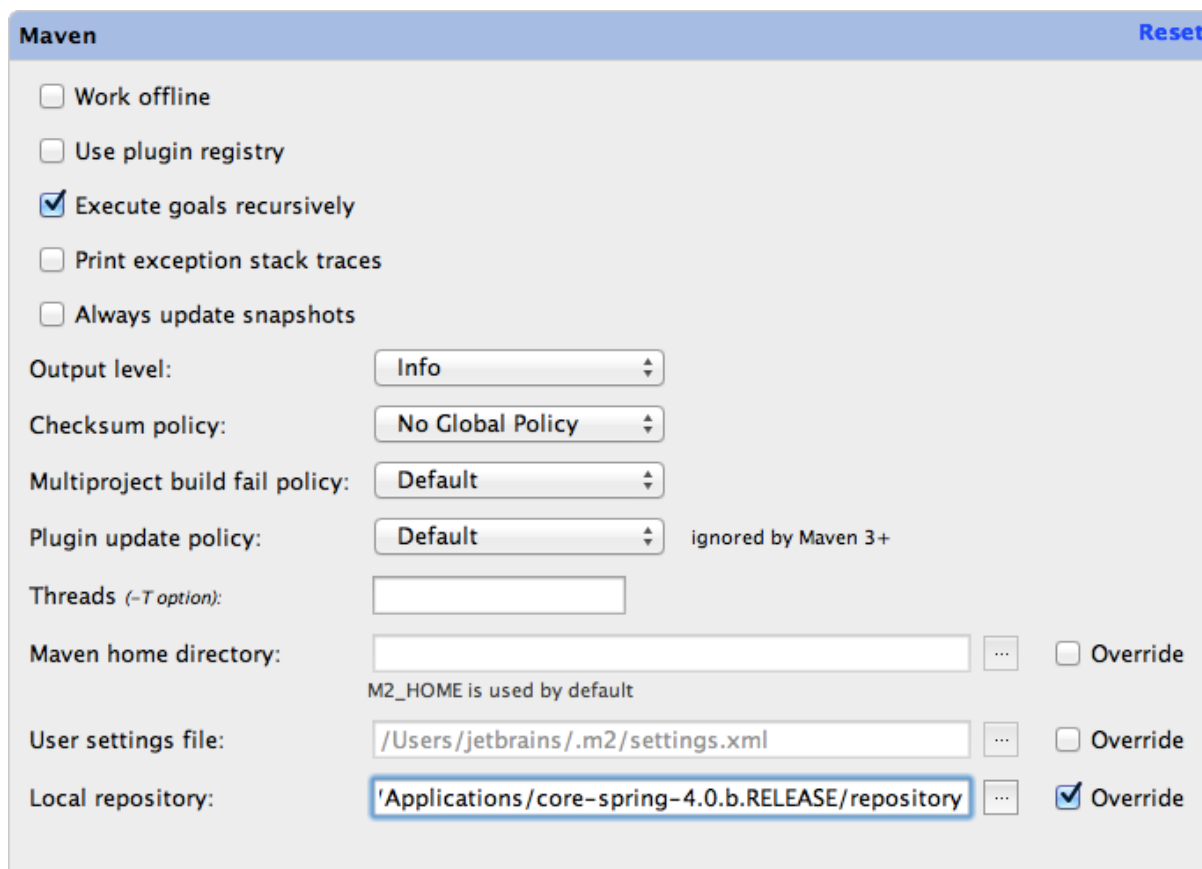
In the *Settings* dialog:

1. Switch to the *Maven* tab;
2. Select the *Override* checkbox next to the *Local repository* setting; and
3. Specify the path to the Maven repository folder, a sub-folder of the course installation folder. The default course installation folder is:

Windows: C:\<course-name>

MacOS: /Applications/<course-name>

Linux: /home/<user-name>/<course-name>



The screenshot shows the 'Maven' settings dialog. The 'Local repository' field is highlighted with a blue box and contains the path '/Applications/core-spring-4.0.b.RELEASE/repository'. The 'Override' checkbox for the local repository is checked. Other settings include 'Work offline' (unchecked), 'Use plugin registry' (unchecked), 'Execute goals recursively' (checked), 'Print exception stack traces' (unchecked), 'Always update snapshots' (unchecked), 'Output level' (Info), 'Checksum policy' (No Global Policy), 'Multiproject build fail policy' (Default), 'Plugin update policy' (Default), 'Threads' (empty), 'Maven home directory' (empty), 'User settings file' (/Users/jetbrains/.m2/settings.xml), and 'Local repository' (/Applications/core-spring-4.0.b.RELEASE/repository).

This example shows `core-spring-4.0.b.RELEASE` on MacOS. The full path is:
`/Application/core-spring-4.0.b.RELEASE/repository`

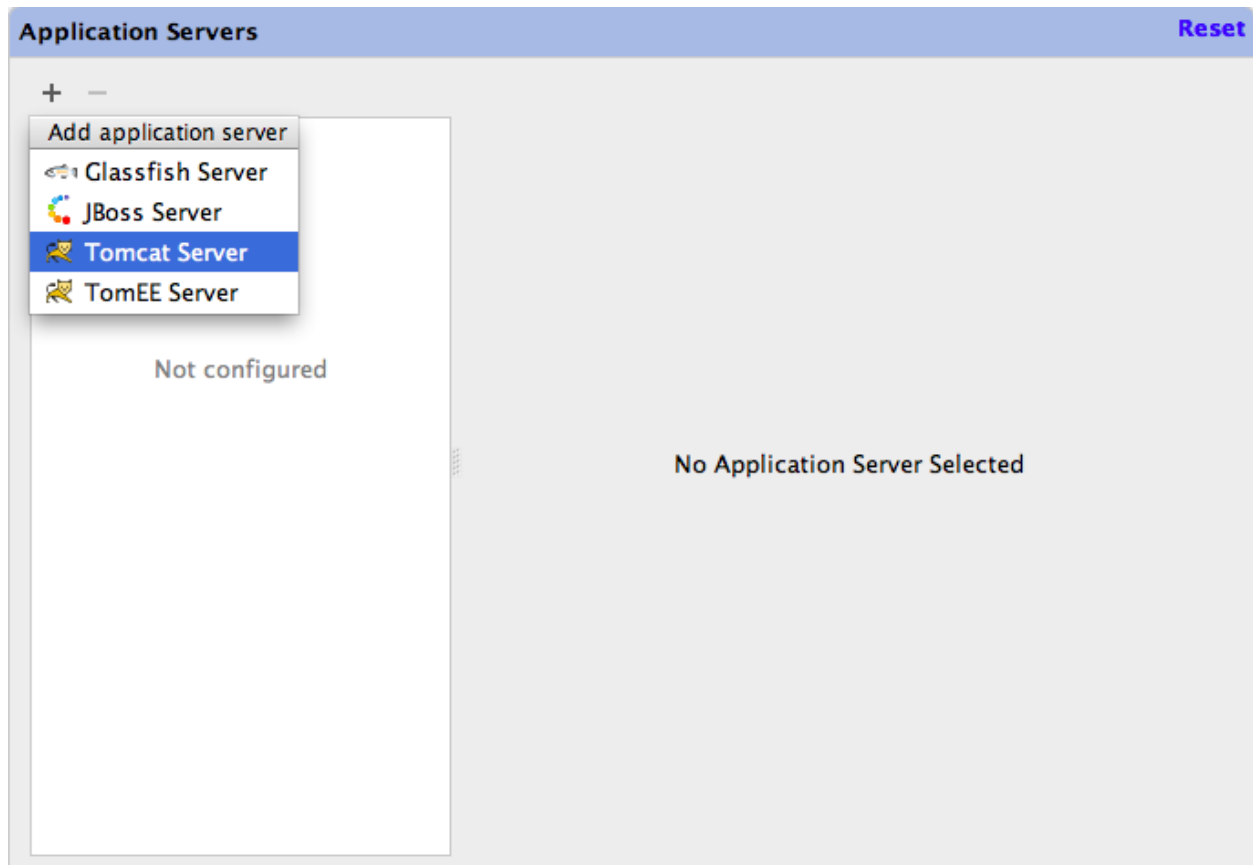
Here are some more examples for different courses and releases:

<i>MS Windows:</i>	C:\spring-web-4.0.a.RELEASE\repository
<i>MacOS:</i>	/Applications/enterprise-spring-4.0.a/repository
<i>Linux:</i>	~/core-spring-4.0.b.RELEASE/repository

Remember this location if you are asked to configure `M2_REPO` later.

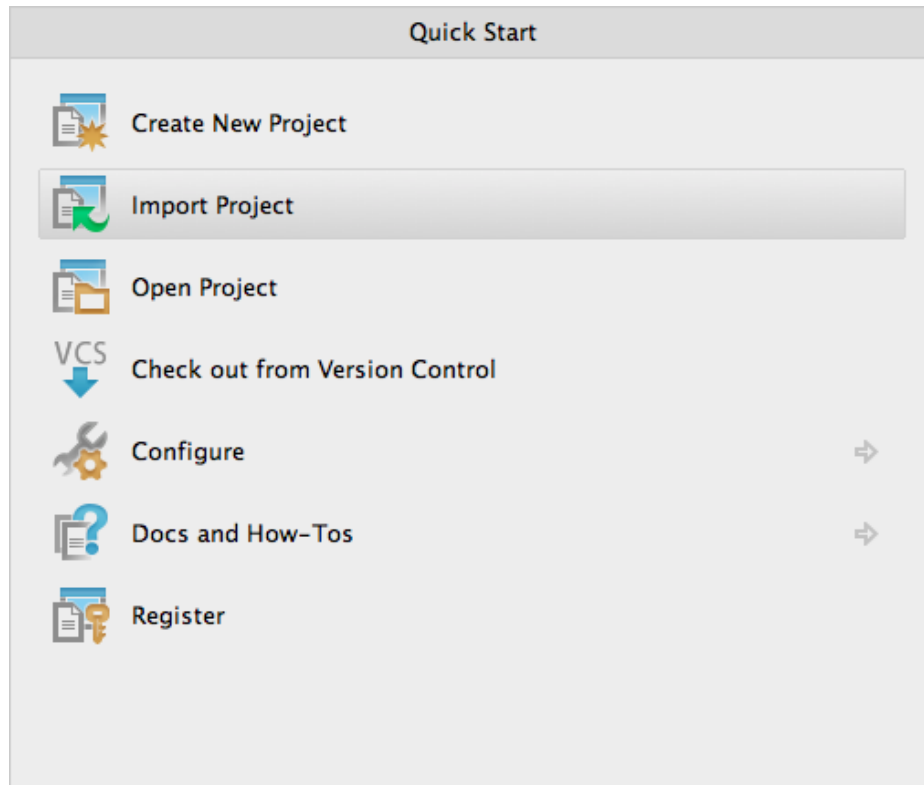
Configure a Tomcat application server

As you will need to run web applications during the course, make sure you've configured a Tomcat application server. To see the list of configured application servers, go back to the *Preferences* dialog and switch to the *Application Servers* tab. Click the plus button to add an application server:



Importing the project into the IDE

To import a project into IntelliJ IDEA click the *Import project* button on the *Welcome screen*:



Importing a Maven project

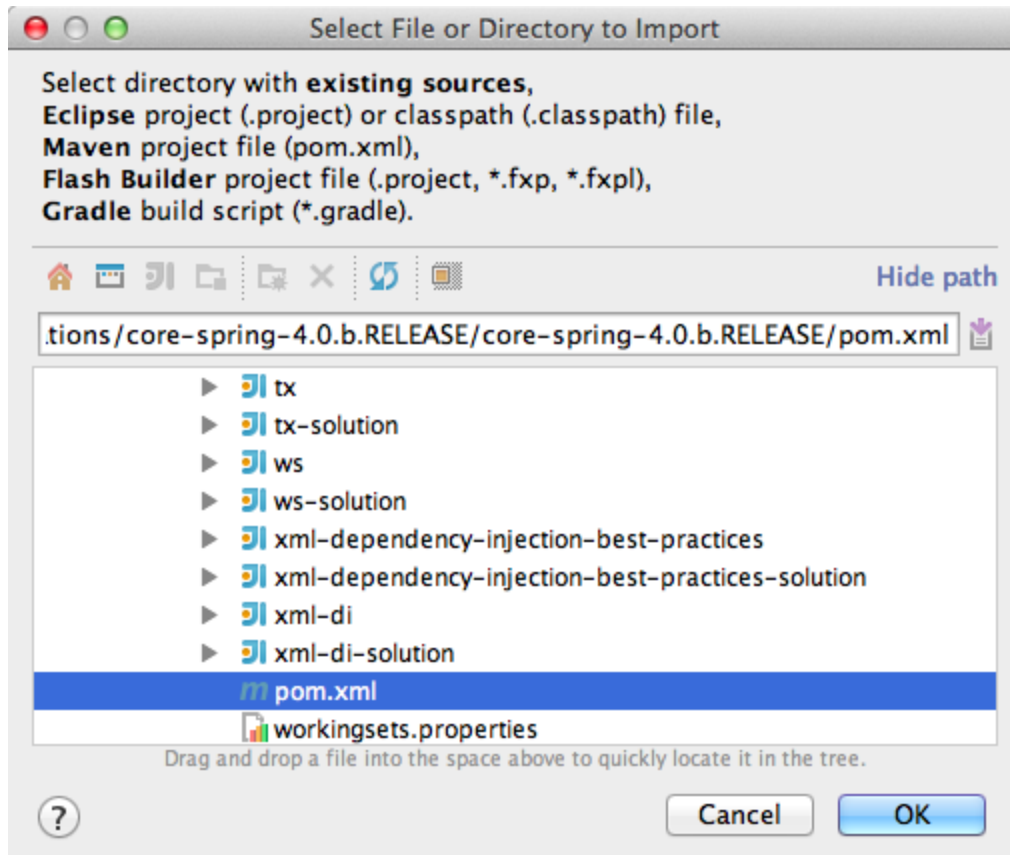
When the project you're trying to import has a root `pom.xml` file (which means this is a Maven project), then you have to choose this `pom.xml` file in the dialog that appears after you've clicked the *Import project* button.

Our courses contain many sub-projects, each with their *own* `pom.xml`. Make sure to pick the *parent POM*, located in the folder that holds all the projects as shown. The parent folder is:

MS Windows:	<code>C:\<course-name>\<course-name></code>
MacOS:	<code>/Applications/<course-name>/<course-name></code>
Linux:	<code>/home/<user-name>/<course-name>/<course-name></code>

The example below shows The example below shows the location of the parent POM for `core-spring-4.0.b.RELEASE` on MacOS. Its full path is:

```
/Application/core-spring-4.0.b.RELEASE/core-spring-4.0.b.RELEASE/pom.xml
```

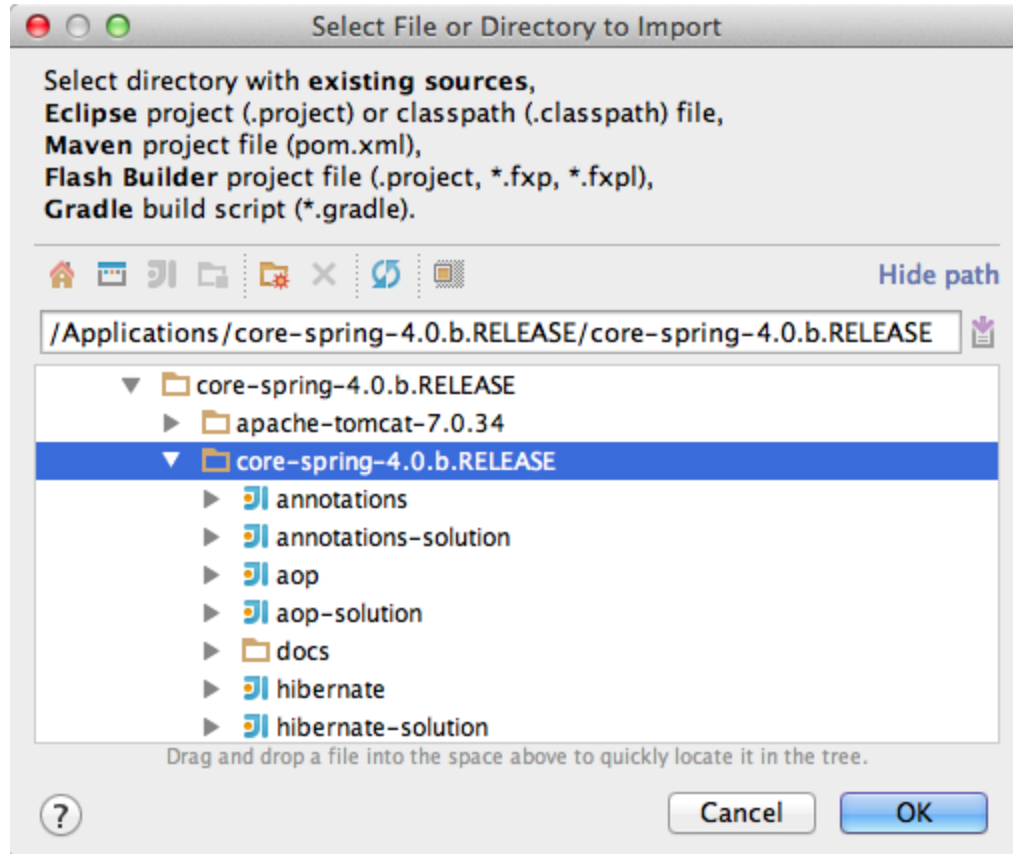


On MS Windows the same file would be at:

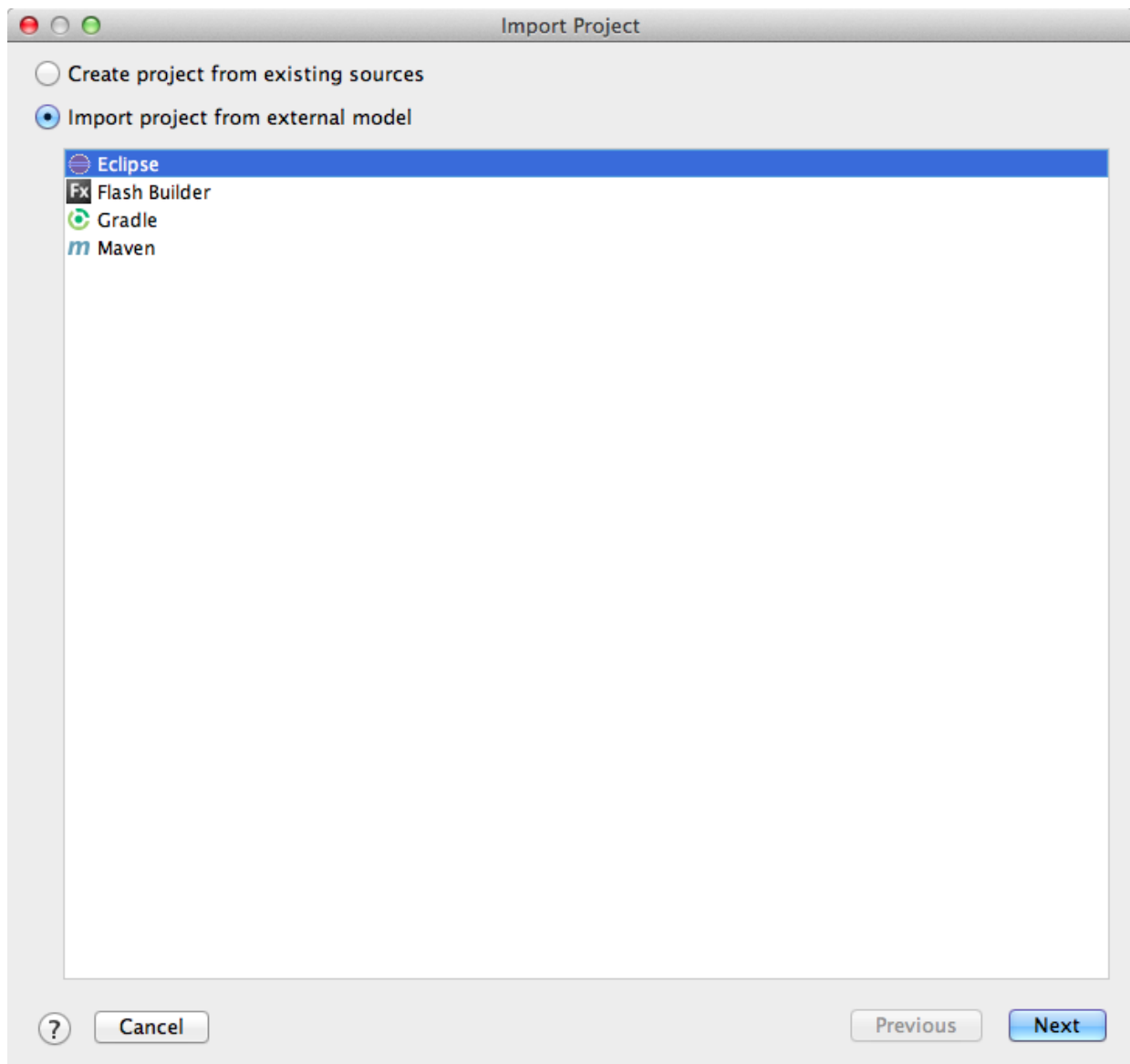
c:\core-spring-4.0.b.RELEASE\core-spring-4.0.b.RELEASE\pom.xml

Importing Eclipse projects

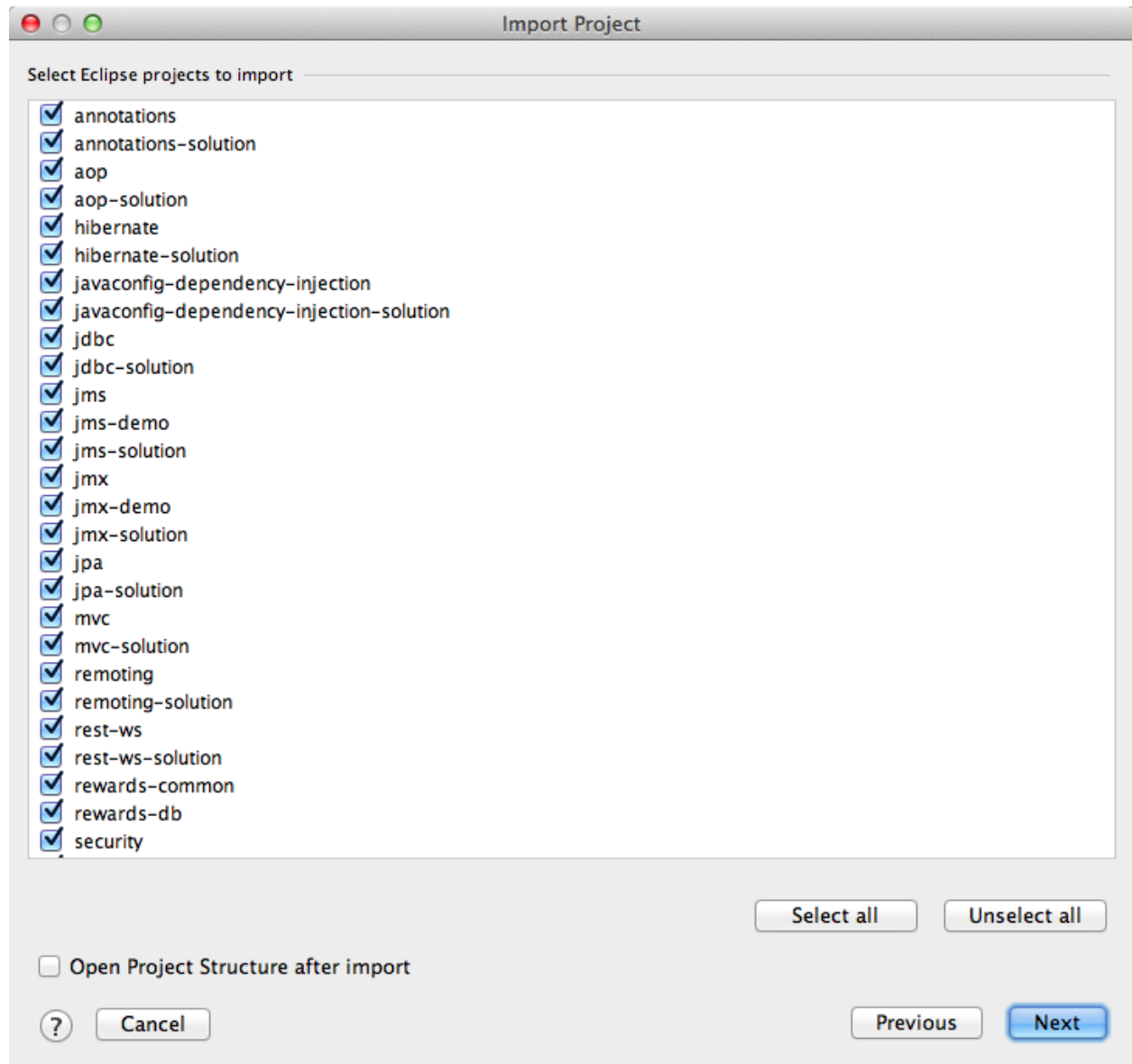
When the project comes with no root pom.xml file you can import it as an Eclipse project using its .classpath file. In our case there is more than one project, so choose the entire root folder:



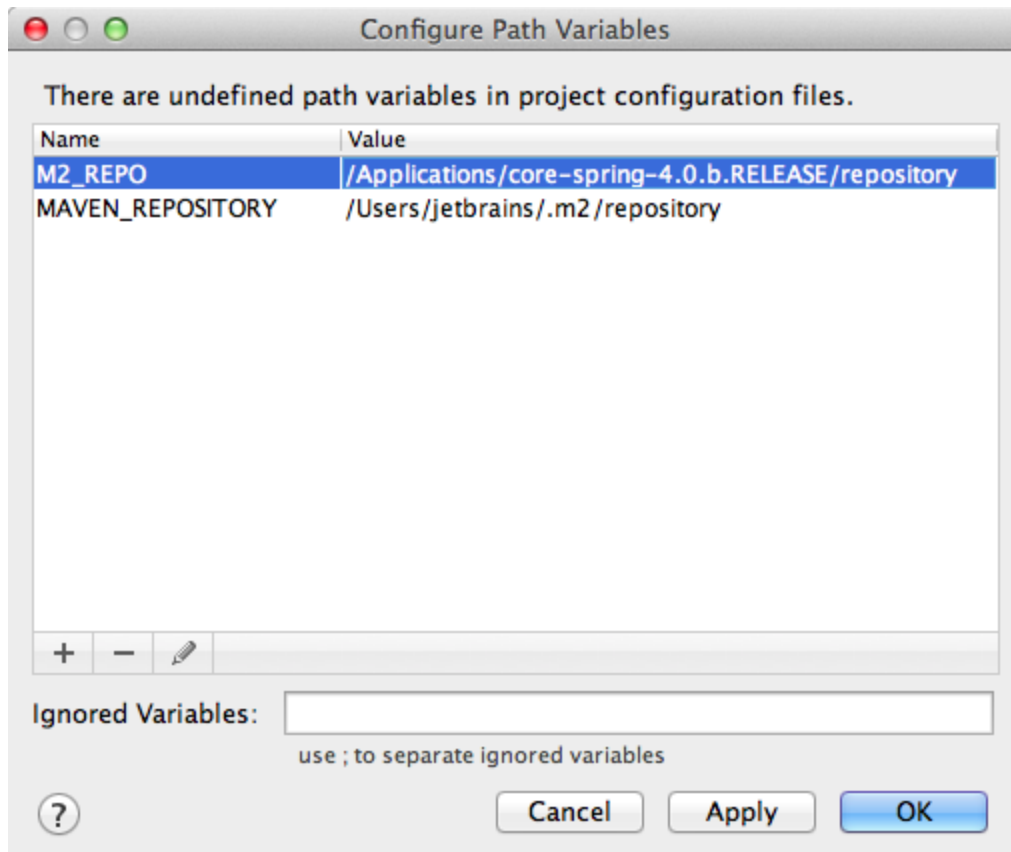
After you've chosen the folder, the IDE will ask you which *external model* to use for the import. Make sure you've selected Eclipse:



If there are several projects in the folder, the IDE will ask you to select the projects to import:



At this point IntelliJ IDEA may prompt you to set the `M2_REPO` variable. Set this variable to point to the `'repository'` folder within the install folder (the same location that we configured earlier). Once set, you should now see all of the Eclipse projects as modules within IntelliJ.

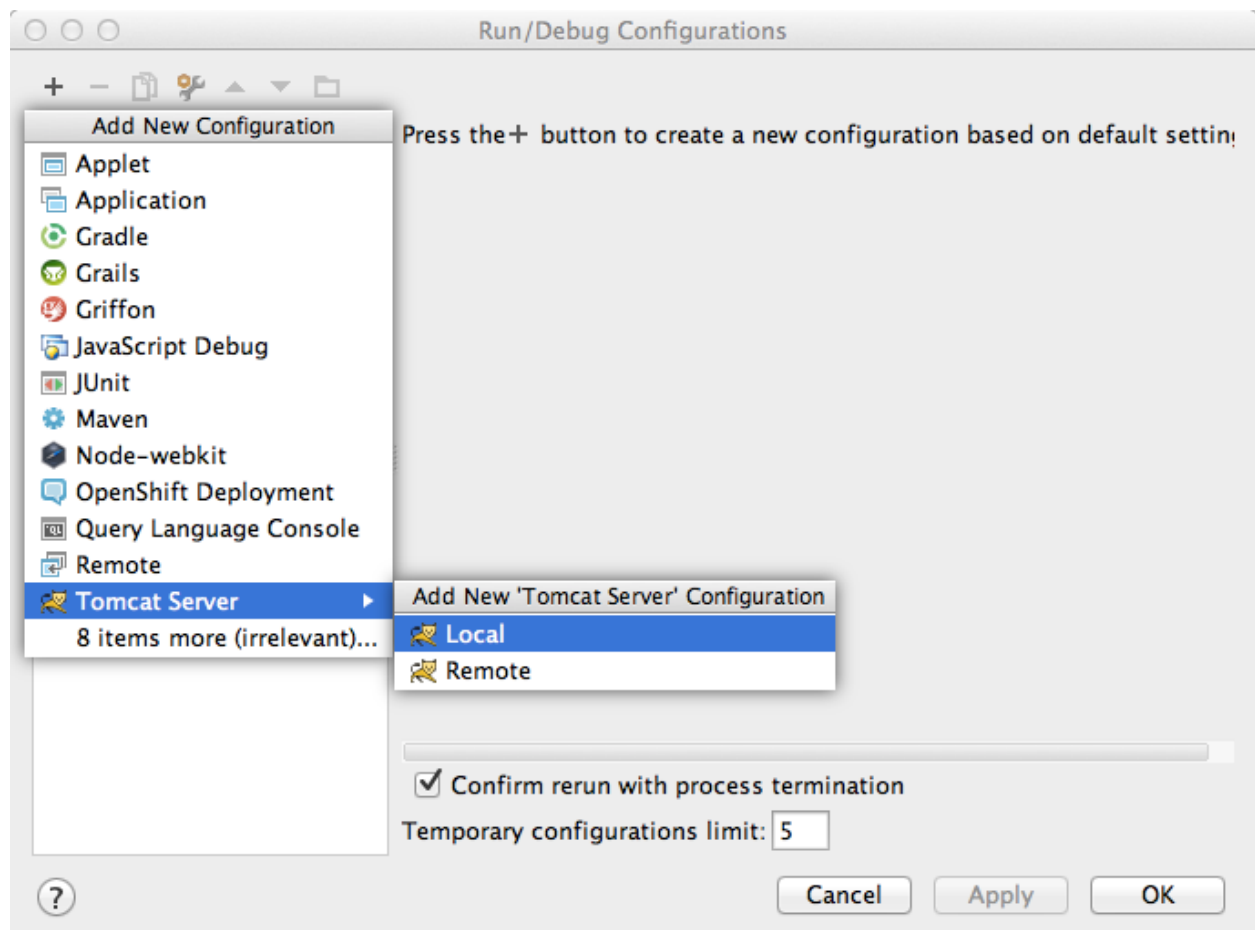


Finally, after the IDE has imported the project, make sure it is compiled without errors. Once the import is finished the IDE may offer you to restart the IDE, please do it.

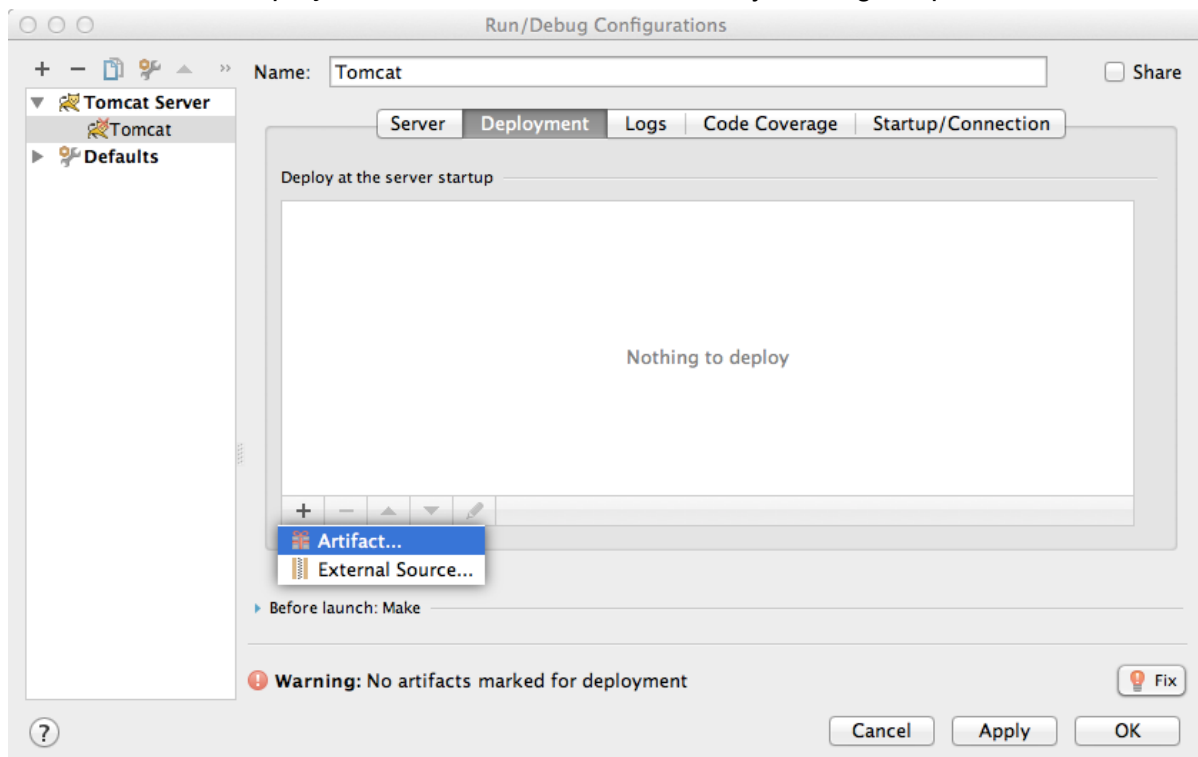
Running applications and tests

Deploying web applications

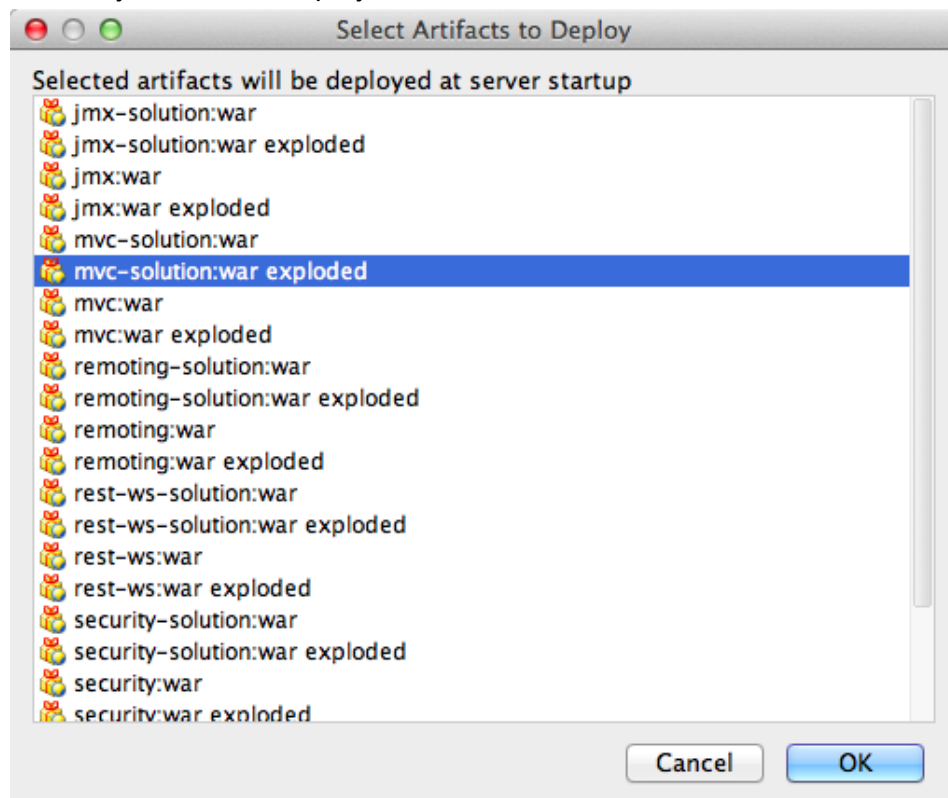
To run a web application, deploy the corresponding artifact to the application server. A *Run configuration* defines how artifacts are deployed to a server. Go to the *Run* → *Edit Configurations* menu, and add a *Local Tomcat* configuration. The *Local* run configuration will start a new instance of the configured server and deploy artifacts there.



Then switch to the *Deployment* tab and add the artifacts by clicking the plus button:

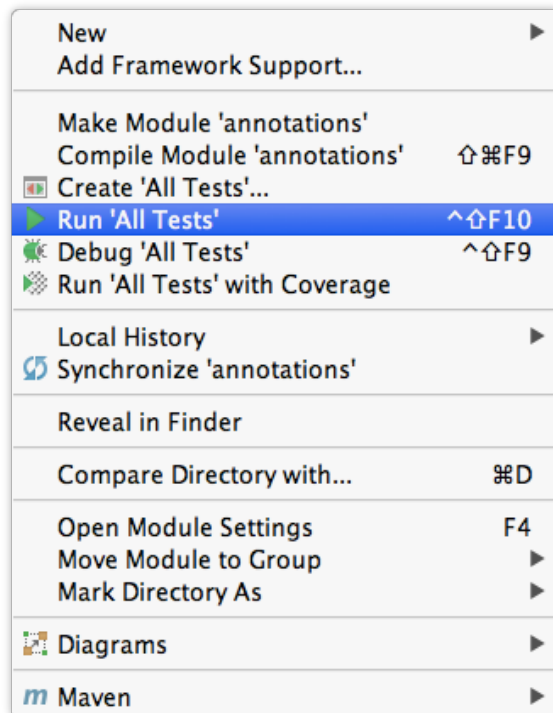


Select the artifacts you'd like to deploy to the server:



Running tests

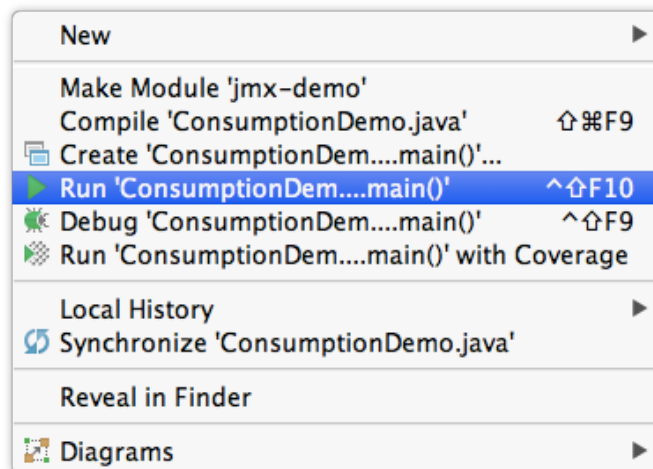
To run all tests from a package or the entire project, simply select *Run 'All Tests'* from the context menu in the *Project* tool window:



If you want to run tests from a single class, use the corresponding action from the context menu for that particular class. To use specific parameters for running tests, you can create a run configuration manually via the *Run* → *Edit Configurations* menu.

Running applications

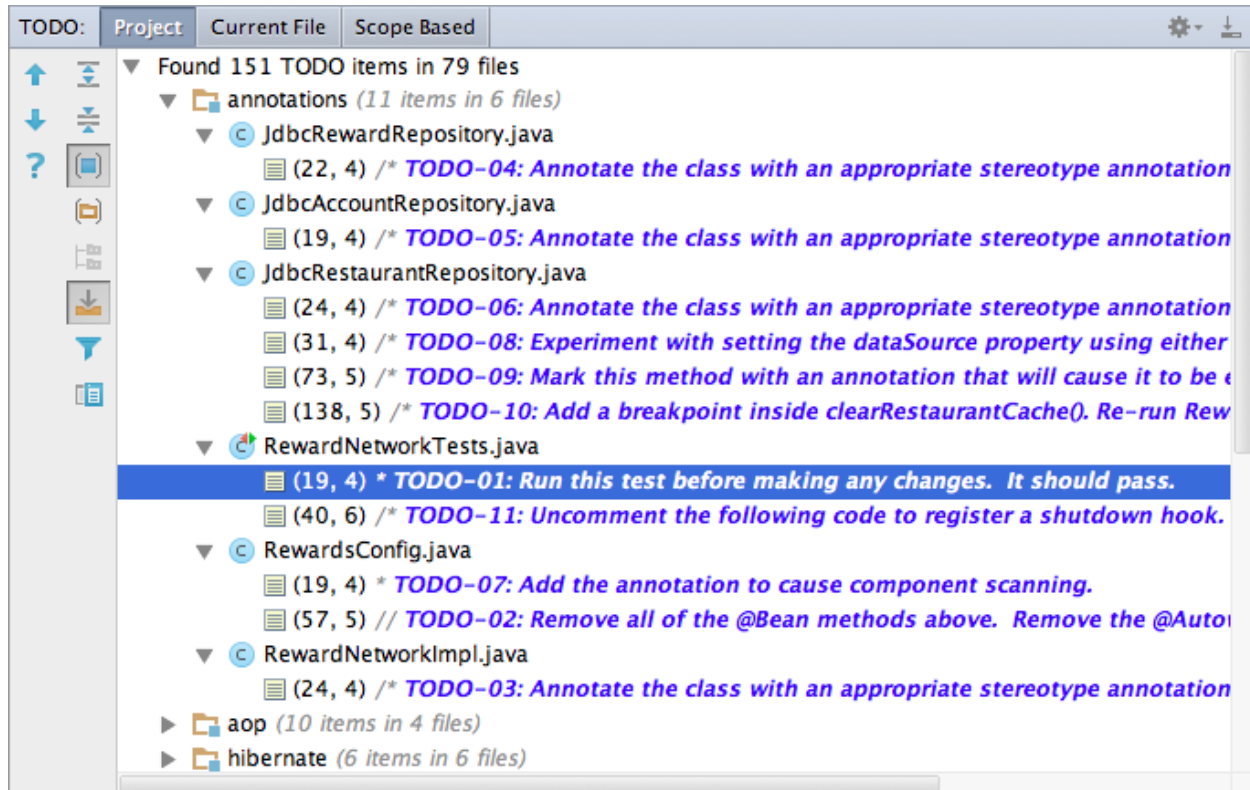
To run an application from its main method, use the corresponding context menu action:



Or create a run configuration manually via the *Run* → *Edit Configurations* menu.

Working with TODOs

To see the list of TODO instructions, use the *TODO* tool window, which can be opened from the left-hand bottom corner of the IDE. Use the toolbar buttons to group items by module:



Other resources

Refer to the following resources to learn more about IntelliJ IDEA:

- [IntelliJ IDEA quick start guide](#)
- [How to migrate from Eclipse to IntelliJ IDEA](#)
- [IntelliJ IDEA help](#)