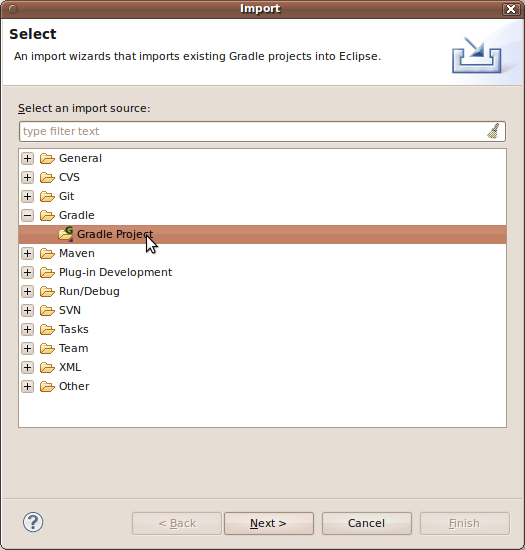
Instalar

desde marketplace

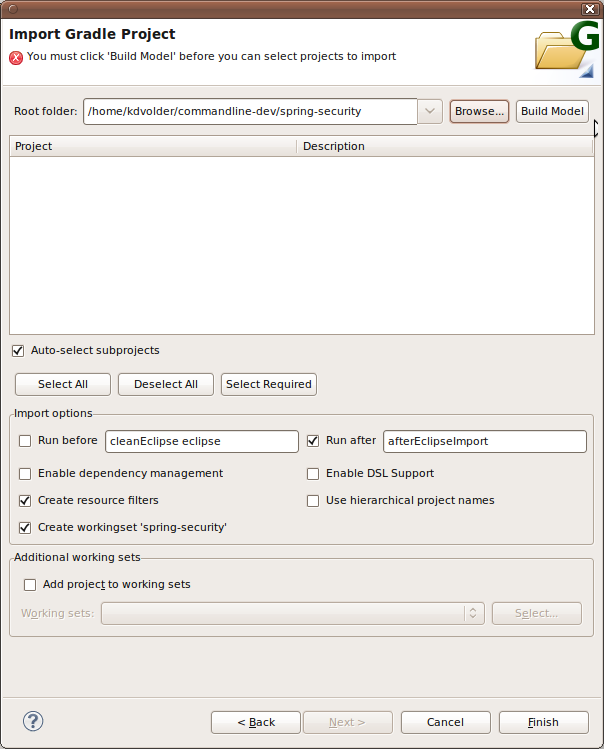
## **Importing Existing Gradle Project(s)**

Now that we have the source code checked out we can use the **Gradle Import Wizard** to import a subset of the spring-security projects into an Eclipse workspace.

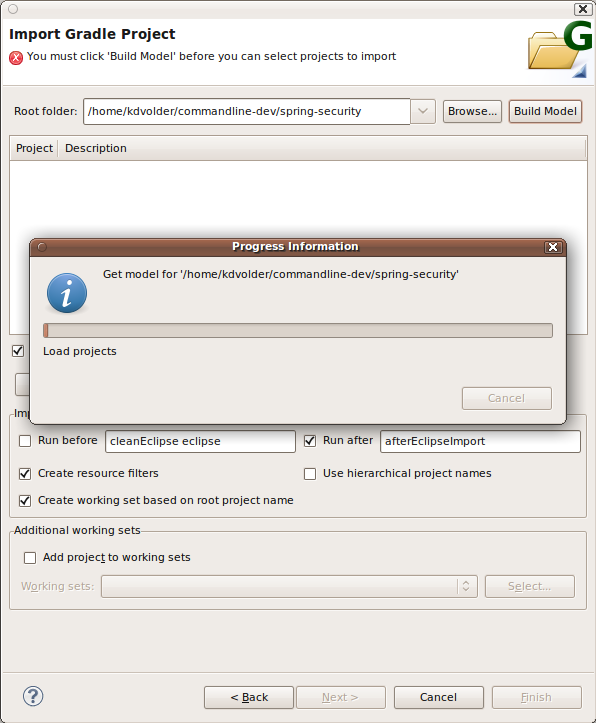
This wizard is accessible via the **"File >> Import ..." menu**.



The first thing you have to do in the wizard is point it to the root folder of your Gradle project. Click the **"Browse" button** and find the folder, or simply enter the path to your project folder if you remember it.



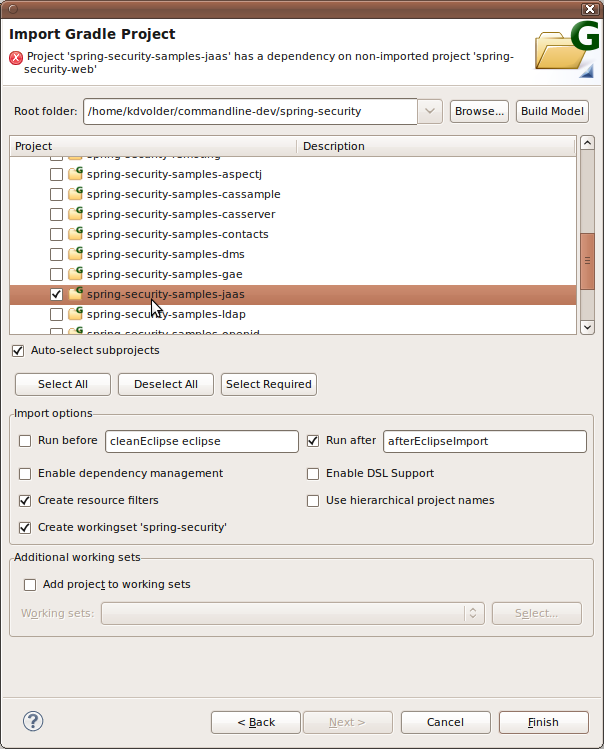
A typical Gradle project may consist of a hierarchy of (sub)projects. Before you can select which (sub)projects you are interested in you must request the Gradle API to construct a Gradle project/sub-project structure model. Click the **"Build Model" button** to do this.



**Warning:** This may take some time, so be patient!

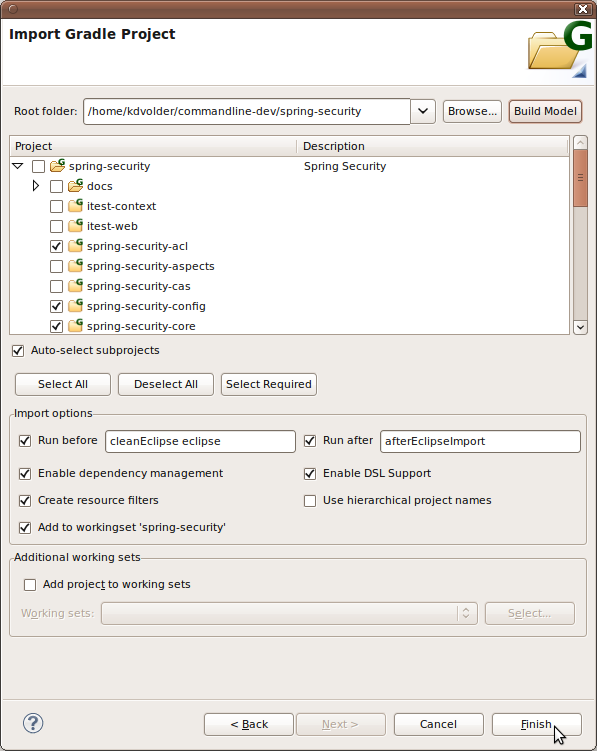
* Building the model may take some time. How much time depends on the complexity of your project and how many of the build script's dependencies can already be found in the cache.
* For spring-security, an initial "Build Model" with a completely cleared .gradle cache may take a few minutes to complete.

After some time, once the model has been constructed, you will be able to select which projects you wish to import into the workspace. You could elect to import everything by clicking the **"Select All" Button**. However, let's assume we are only interested in the "spring-security-samples-jaas" project.



## **Performing the Import**

For this tutorial we leave everything at the default settings, except for the 'run before' option which we explicitly enable since we want to have correct configuration for the WTP aspects of the sample web project.

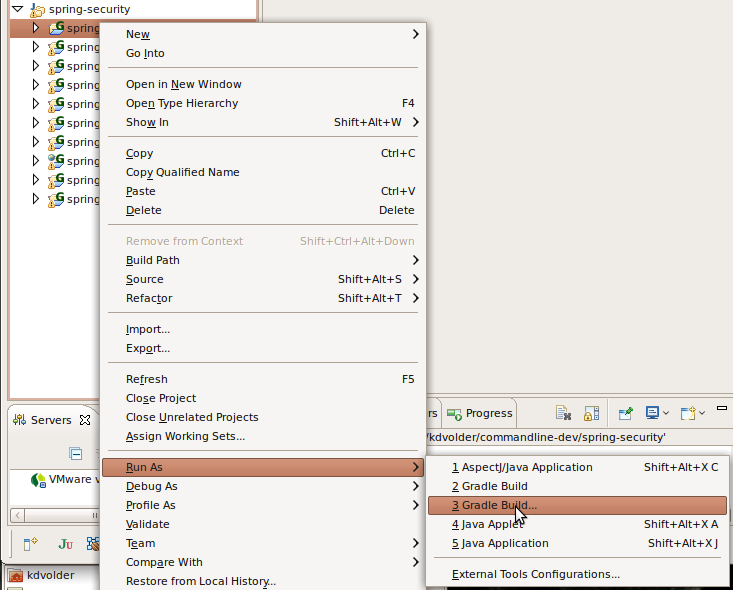


## **The Task UI: "Launching" Gradle Tasks**

In the next section of this tutorial we'll walk through the UI for creating and launching Gradle launch configurations. Gradle launch configurations are provided as extensions to the standard Eclipse launching framework. This means that if you are already familiar with the Eclipse launch UI, most of this will be familiar to you.

**Creating and Executing a Launch Configuration**

Before we can "launch" a list of Gradle tasks, we must create a launch configuration. The easiest way to do that is by using the **"Run As" context menu.** This menu is accessed by right clicking on a Gradle project or one of its nested resources in the Package Explorer.



* The following menu items will be shown (others may also be shown depending on what exactly you right-clicked on).
* **Run As >> Gradle Build ...** (dots at the end)
* This opens a Gradle launch configuration associated with the current project in the Tabbed launch configuration editor. This allows you to examine and edit the configuration before launching it.
* If no launch configuration exists yet (for the selected project) an empty one will be created automatically. If more than one configuration is found, you will be asked to choose one.
* **Run As >> Gradle Build** (no dots at the end)
* This attempts to directly launch a Gradle launch configuration associated with your project. Typically you would use this if you already have a launch configuration defined and want to just launch it without examining or editing it. If a launch configuration doesn't exist yet, however, one will be created and opened in the editor. If more than one configuration already exists for the selected project, then a dialog will popup to ask you to choose one.
* **External Tools Configurations**
* This opens the generic Eclipse "External Tools Configurations" editor, allowing you to create / edit / save etc. launch configurations of different types, including Gradle launch configurations.