KPME Developer Setup Guide V2

Introduction

Welcome to KPME! If this is your first time setting up your environment, please ensure that you have administrator access to the computer, otherwise some portions of this guide may not work.

This page extends on KPME Developer Setup Guide, to include instructions for setting up KPME versions 2.0+. If you are setting up a workspace for projects **earlier** than V2.0, please refer to the original setup guide KPME Developer Setup Guide.

Software Requirements

The following is a list of software that is essential to install in order to get a working KPME development environment up and running.

- JDK (Recommended JDK 7u55 64bit)
- Apache Maven 3.0.4
- MySQL 5.1 or above
- Eclipse + Subversive with required dependencies

Installation and Configuration

Follow the steps outlined by each bold heading below in order to configure the default development stack.

Install the JDK

- Download and install the latest Java 7 JDK. The 64bit version is preferred since it will be able to address more memory for your environment, but if you are limited on space, the 32bit version will also work.
- Add `JAVA_HOME` to your environment variables and set it to the base directory of your installation (Example: `C:\Java\jdk1.7.0_25`).
- Add `JAVA_HOME/bin` to your path.

Install Apache Maven

- Download and install the latest Apache Maven 3.
- Add `M2_HOME` to your O.S. environment variables and set it to the base directory of your installation (Example: `C:\Java\apache-maven-3.0.4`).
- Add `M2_HOME/bin` to your path.

Note: You MUST use the variable name M2_HOME. The launch configuration files reference this environment variable at run-time, and will give you a class load error if it is not defined correctly in your environment.

- (Optional) If you wish to change the location of your repository to something other than your home folder, create a file named "settings.xml" in USER_HOME/.m2 with the following contents, replacing the path with your preferred location:

```
<settings>
  <localRepository>your/custom/path/to/repository</localRepository>
</settings>
```

Install a Database

The default database for KPME is MySQL 5.1 or above. While Oracle is also supported in releases, it is untested and therefore not ready for developers.

- Download and install MySQL Server Community Edition 5.1 or above. Make sure to select a **Detailed Configuration**and enable the following options:
  - Developer Machine
  - Multifunctional Database
  - Best Support for Multilingualism
- Install MySQL Workbench to manage and view your database. **Note: This may require the installation of additional software, such**

as VC++ and .NET packages.

- If you are on Linux or MacOSX, you need to set a parameter to force MySQL to store tables in lower case. To do this, go to `/etc/my.conf`and make sure the following is set:

```
lower_case_table_names=1
```

Configuring the Database

The easiest way to set up databases and permissions is to do it via the command line, however you may wish to do this via your MySQL Workbench. Developers typically use two databases for their server and one database for their unit tests. The following table lists the default settings for these databases that will give the most minimal setup.

	Schema/Name	**Username**	**Password**
KPME Client Database	tk	tk	tk_tk_tk
Rice Server Database	krtt	krtt	krtt
Unit Test Database	tk_test	tk	tk_tk_tk

To configure the database from the command line, use the root user name and password you created during MySQL Server installation to log into your database server, then issue the following two commands for each row in the above table. Replace *dbname* with the entries from column "**Schema/Name**" and *username* and *password* from the corresponding **Username** and **Password** columns.

```
CREATE DATABASE dbname CHARACTER SET utf8 COLLATE utf8_bin;
GRANT ALL ON dbname.* TO 'username'@'%' IDENTIFIED BY 'password';
```

To configure the database using MySQL Workbench:

- Open MySQL Workbench
- Go to **SQL Development**and double-click on your local instance. For each database above:
  - Right-click on the left bar and select **Create Schema...**
  - Name it as above and select **utf8 - utf8_bin** for the collation. Apply these changes.
- Go to **Server Administration**and double-click on your local instance. For each database above:
  - Go to the **Accounts** tag and click on the **Add Account** button.
  - Add a user with the corresponding username and password and **Apply**.
  - Go to the **Schema Privileges** tag and select the database you are editing.
  - Click **Add Entry...**
  - Under the **Schema** section, check **Selected Schema,** select `localhost` and select the database you are editing, and click **OK**.
  - Click the **Select "ALL"** button and save your changes.

Intellij or Eclipse (choose one or both)

Install the Eclipse IDE

Install the Eclipse IDE

The default development environment for KPME is Eclipse 3.5 or above. Any flavor should be ok, but it is recommended to start out with the Eclipse IDE for Java Developers since it is small.

- Download and install Eclipse 3.5 or above.
- Go to your install directory and edit `eclipse.ini`. You will need to add your JDK and other memory arguments to make sure your environment will run smoothly. The following has the recommended defaults but you may need to increase them in order to match your machine.

```
-vm
C:/Program Files/Java/jdk1.7.0_25/bin/javaw.exe
-vmargs
-Xms1024m
-Xmx1024m
-server
-XX:+CMSClassUnloadingEnabled
-XX:+CMSPermGenSweepingEnabled
-XX:+UseConcMarkSweepGC
-XX:PermSize=128m
-XX:MaxPermSize=128m
```

- Start eclipse and choose your preferred workspace.
- Verify that you are working on Java 6 by going to **Window -> Preferences -> Java -> Installed JREs**. Make sure that jre6 is selected.
- Disable validation by going to **Preferences -> Validation** and checking **Suspend All Validators**.
  NOTE: Disabling Validations in this way does not seem to be working in Eclipse Kepler. It causes the kpme-web project build to hang. If you have problems with the build taking a very long time and not building, please see solution in the Troubleshooting section at the bottom of this page.

Install Subversive

- Click **Help -> Install New Software**.
- In the **Work with** drop down menu, select the `http://download.eclipse.org/releases` entry that matches your Eclipse version. Go to Collaboration and select all required Subversive entries.
- Click **Finish** and continue through all the screens in order to install your software.
- Restart Eclipse when promped.

Install M2Eclipse (Optional)

Note: Only complete this step if using Eclipse 3.6 or below. Eclipse 3.7 and above have switched to an internal Maven implementation, m2e.

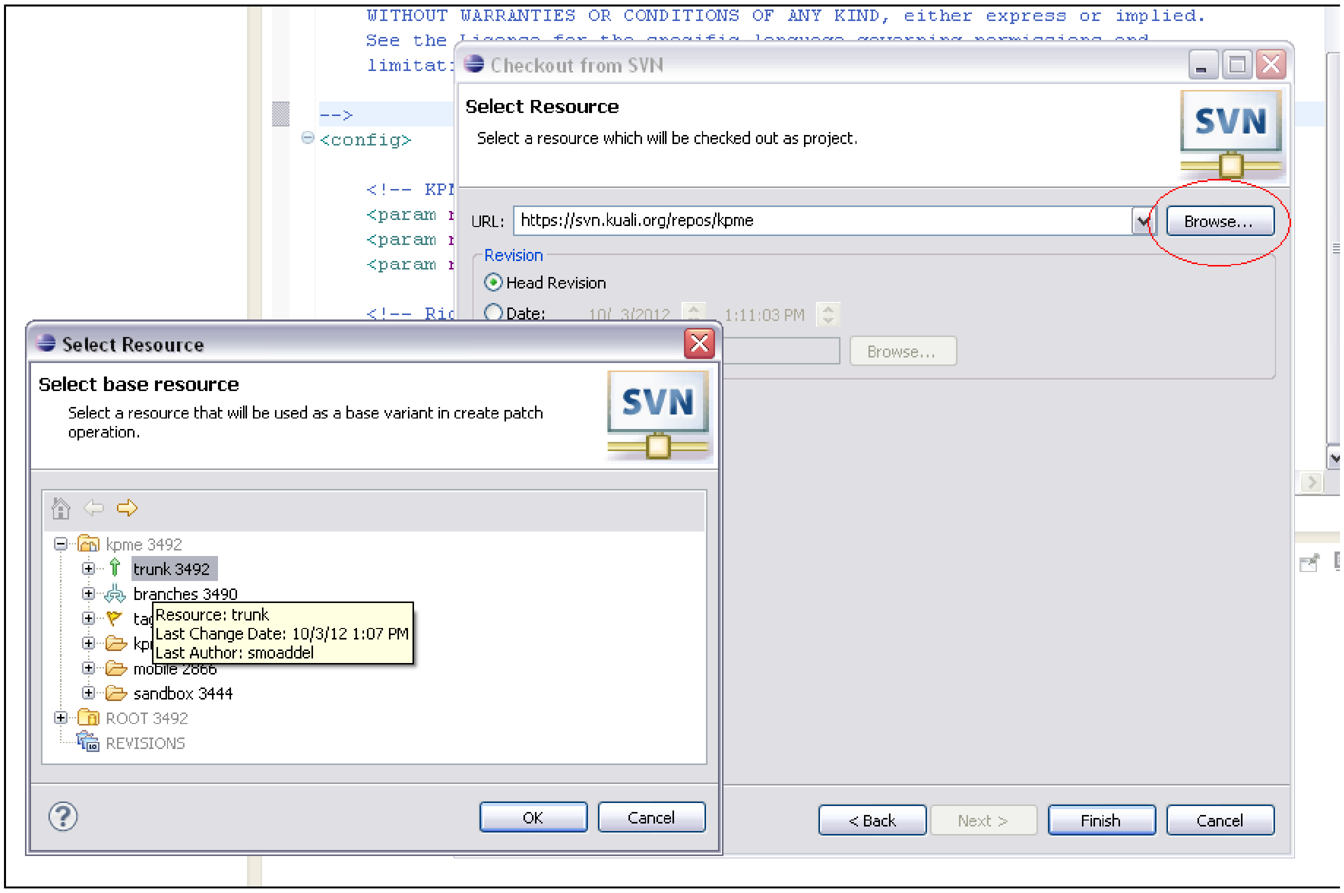
- **Help -> Install New Software**.
- Click **Add** and add the location http://m2eclipse.sonatype.org/sites/m2e..
- Select your new repository from the drop down list.
- Find and select **Maven Integration for Eclipse.**
- Click **Finish** and continue through all the screens in order to install your software.
- Restart Eclipse when prompted.

Create the KPME Maven Project in Eclipse

You will be loading the project into your workspace from a repository via the Subversive plugin.

Checkout and Configure the Project

- In Eclipse, right click in the **Package Explorer** window and select **New -> Other -> SVN -> Project from SVN**.
- Select the option to **Create a new repository location** and select **Next**.
- Enter the URL https://svn.kuali.org/repos/kpme and other authentication information. Select **Next** to continue.
- Select "Browse" to bring up a dialog box where you can select the branch you will be working on. Most developers will be on trunk, so just select `trunk` if you are not given any additional instructions.


WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
Checkout from SVN
Select Resource
Select a resource which will be checked out as project.
URL: https://svn.kuali.org/repos/kpme
Browse...
Revision
Head Revision
Date:
Select Resource
Select base resource
Select a resource that will be used as a base variant in create patch operation.
kpme 3492
trunk 3492
branches 3490
Resource: trunk
Last Change Date: 10/3/12 1:07 PM
Last Author: smoaddel
mobile 2866
sandbox 3444
ROOT 3492
REVISIONS
OK
Cancel
< Back
Next >
Finish
Cancel

- Click **Okay**, then click **Finish**.
- When prompted how you want to check out your project, select **Check out as a project with the name specified** and make sure the name is `kpme`. Click **Finish** to begin the install process.

At this point, Subversive for Eclipse will begin the checkout process for KPME and create a new project in your workspace. When the checkout process is complete, do the following

- Right click your new `kpme` project and select **Maven -> Enable Dependency Management**.
- **Note: If you are not given a menu entry for "Maven", you must convert the project to a maven project.** Right click on the project folder and select **Configure -> Convert to Maven Project**

Maven for Eclipse will download all the jars and files that are listed in the project POM's dependencies.

Install the Kuali Configuration File

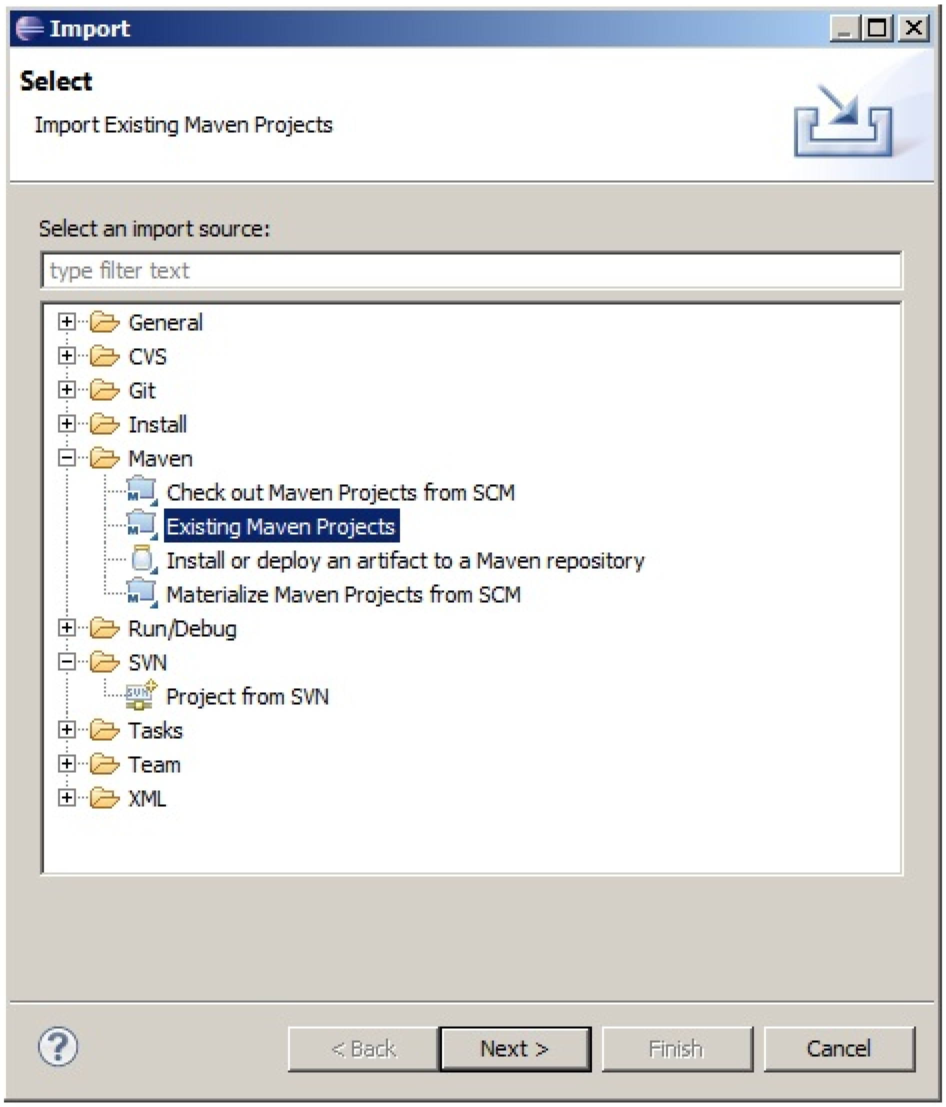
In your new project, go to `env/src/main/config` and copy the config files in the folder into your `USER_HOME`.

- The main configuration file kpme-config.xml for your server environment should be `in USER_HOME/kuali/main/dev/kpme-config.xml`.
- The test configuration file kpme-test-config.xml for your unit tests should be in `USER_HOME/kuali/test/dev/kpme-test-config.xml`.

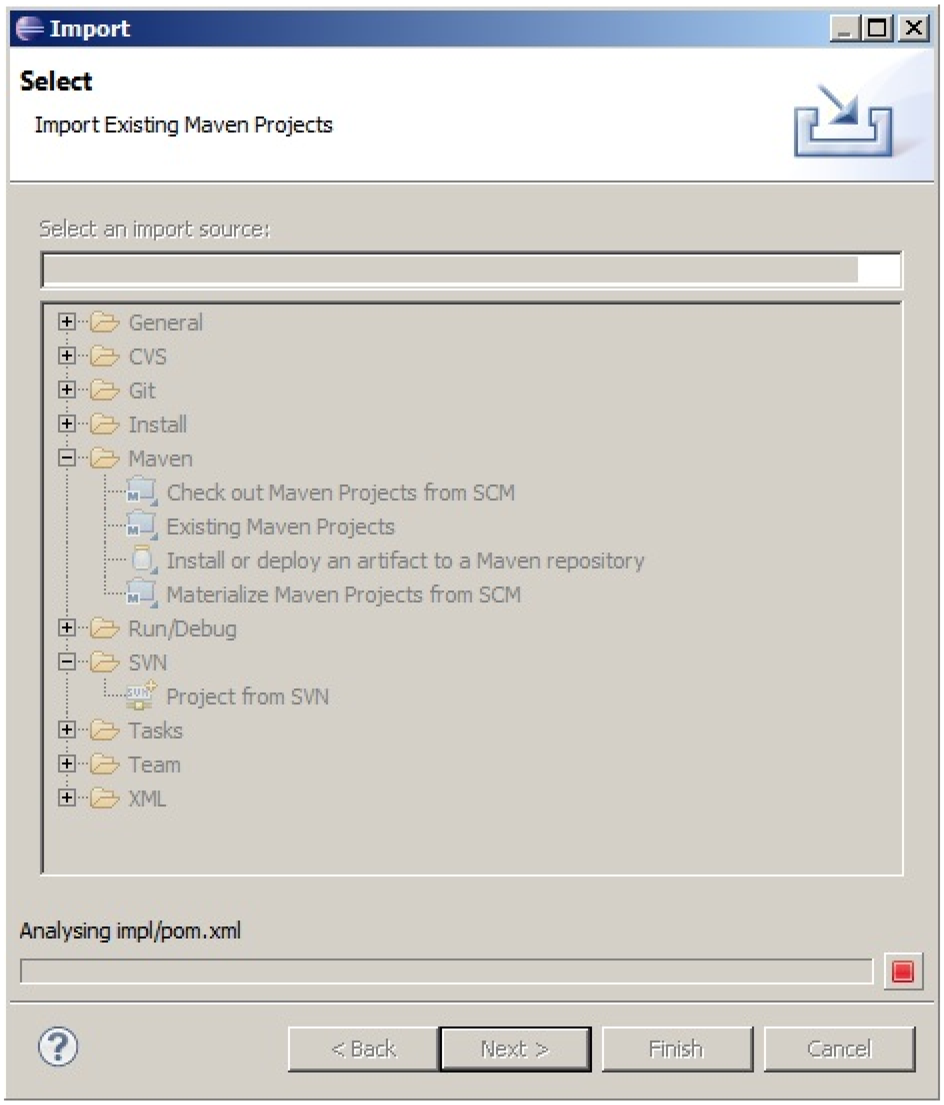
Import existing maven projects

Now that we have our base project imported into our workspace, we also need to import each of the projects sub modules. This import will not create duplicate projects folders on disk, but rather derive themselves from the existing project.

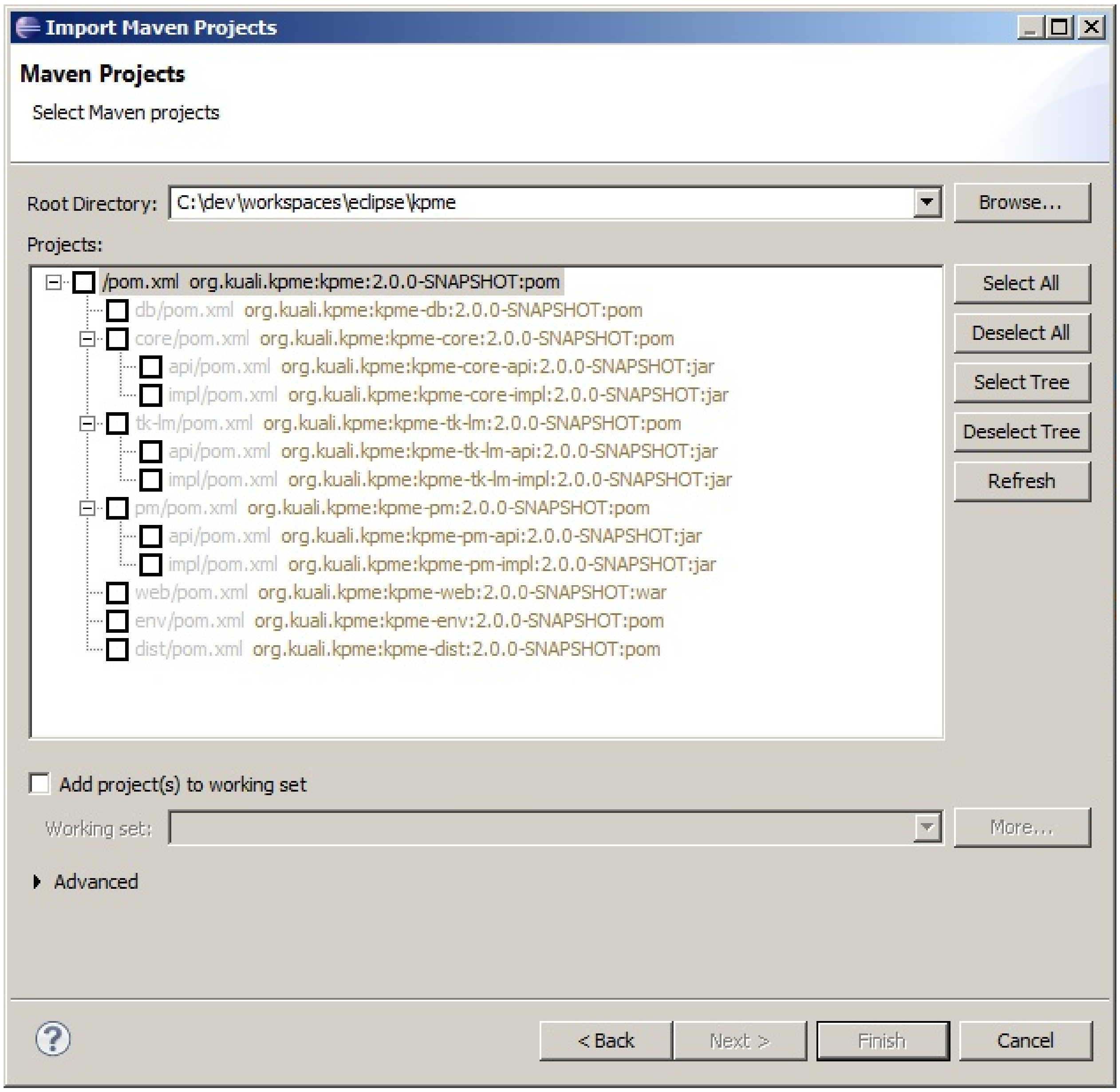
- In the package explorer view, right click on the project created in the previous sections and select "Import".
- In the dialog box that pops up, expand "Maven" and select "Import Existing Maven Project".



- If the correct project was right-clicked, eclipse's maven should begin scanning the project tree for buried poms. If not, you may need to browse for the location manually (example: kpme/core), or escape the dialog and try again.



- Once the analysis is complete, the "Import Maven Projects" dialog box should appear, populated with all poms under the project.


Import Maven Projects

Maven Projects

Select Maven projects

Root Directory: C:\dev\workspaces\eclipse\kpme

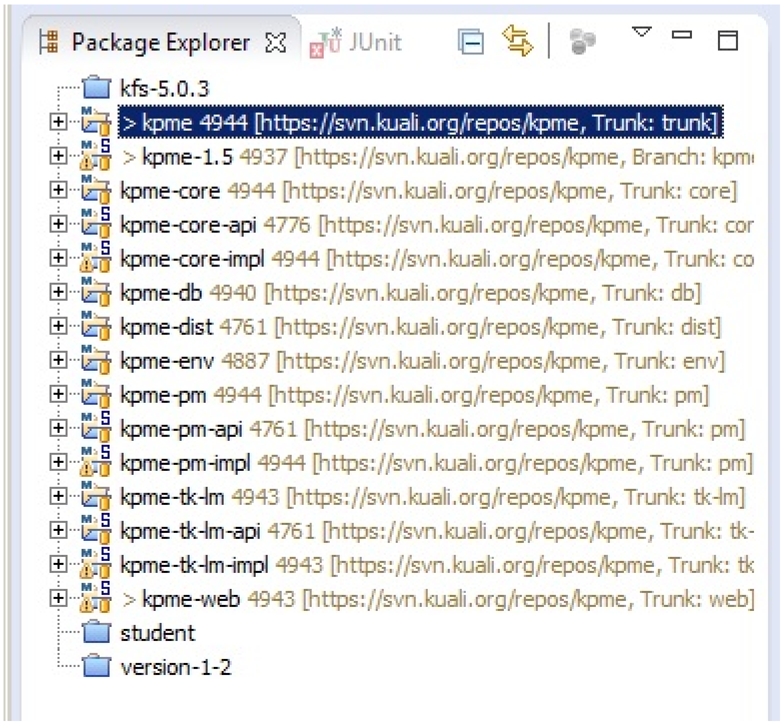
Projects:

```
/pom.xml  org.kuali.kpme:kpme:2.0.0-SNAPSHOT:pom
  db/pom.xml  org.kuali.kpme:kpme-db:2.0.0-SNAPSHOT:pom
  core/pom.xml  org.kuali.kpme:kpme-core:2.0.0-SNAPSHOT:pom
    api/pom.xml  org.kuali.kpme:kpme-core-api:2.0.0-SNAPSHOT:jar
    impl/pom.xml  org.kuali.kpme:kpme-core-impl:2.0.0-SNAPSHOT:jar
  tk-lm/pom.xml  org.kuali.kpme:kpme-tk-lm:2.0.0-SNAPSHOT:pom
    api/pom.xml  org.kuali.kpme:kpme-tk-lm-api:2.0.0-SNAPSHOT:jar
    impl/pom.xml  org.kuali.kpme:kpme-tk-lm-impl:2.0.0-SNAPSHOT:jar
  pm/pom.xml  org.kuali.kpme:kpme-pm:2.0.0-SNAPSHOT:pom
    api/pom.xml  org.kuali.kpme:kpme-pm-api:2.0.0-SNAPSHOT:jar
    impl/pom.xml  org.kuali.kpme:kpme-pm-impl:2.0.0-SNAPSHOT:jar
  web/pom.xml  org.kuali.kpme:kpme-web:2.0.0-SNAPSHOT:war
  env/pom.xml  org.kuali.kpme:kpme-env:2.0.0-SNAPSHOT:pom
  dist/pom.xml  org.kuali.kpme:kpme-dist:2.0.0-SNAPSHOT:pom
```

☐ Add project(s) to working set

Working set:

- Select all projects listed in the dialog box and click "Finish". This may take time, as the projects are imported and then converted to their maven form.



kfs-5.0.3
> kpme 4944 [https://svn.kuali.org/repos/kpme, Trunk: trunk]
> kpme-1.5 4937 [https://svn.kuali.org/repos/kpme, Branch: kpm
kpme-core 4944 [https://svn.kuali.org/repos/kpme, Trunk: core]
kpme-core-api 4776 [https://svn.kuali.org/repos/kpme, Trunk: cor
kpme-core-impl 4944 [https://svn.kuali.org/repos/kpme, Trunk: co
kpme-db 4940 [https://svn.kuali.org/repos/kpme, Trunk: db]
kpme-dist 4761 [https://svn.kuali.org/repos/kpme, Trunk: dist]
kpme-env 4887 [https://svn.kuali.org/repos/kpme, Trunk: env]
kpme-pm 4944 [https://svn.kuali.org/repos/kpme, Trunk: pm]
kpme-pm-api 4761 [https://svn.kuali.org/repos/kpme, Trunk: pm]
kpme-pm-impl 4944 [https://svn.kuali.org/repos/kpme, Trunk: pm]
kpme-tk-lm 4943 [https://svn.kuali.org/repos/kpme, Trunk: tk-lm]
kpme-tk-lm-api 4761 [https://svn.kuali.org/repos/kpme, Trunk: tk-
kpme-tk-lm-impl 4943 [https://svn.kuali.org/repos/kpme, Trunk: tk
> kpme-web 4943 [https://svn.kuali.org/repos/kpme, Trunk: web]
student
version-1-2

You'll notice the disdain in the workspace with all the newly imported projects. With eclipse, we can employ the use of working sets to organize our workspace.

Working Sets (this step is optional)

Optionally you can set up Working Sets in Eclipse using the steps described here:

Configure Working Sets

Starting the Application Server

- The scripts to launch the application version 2.* reside in the in the kpme-env module. This module also contains the local Kuali configuration files needed for start-up.
- In Eclipse, expand the kpme-env project, and go to `src/main/scripts`, right-click on the launch file KPME Jetty 7.launch and select **Run As...**and the file name.
- If all goes well, the server will start up, populate your database, and then launch.
- Once you get the message that the server has started, open a browser and navigate to http://localhost:8080/kpme-dev/Time.do.

Troubleshooting

Error: Could not find or load main class org.codehaus.classworlds.Launcher

- At a command prompt:

```
>% mvn --version
```

- You should receive output that displays version information about your maven installation. If you do not, verify that you have added the variable name M2_HOME, and added it to your path.

Javascript Validation causing build to hang

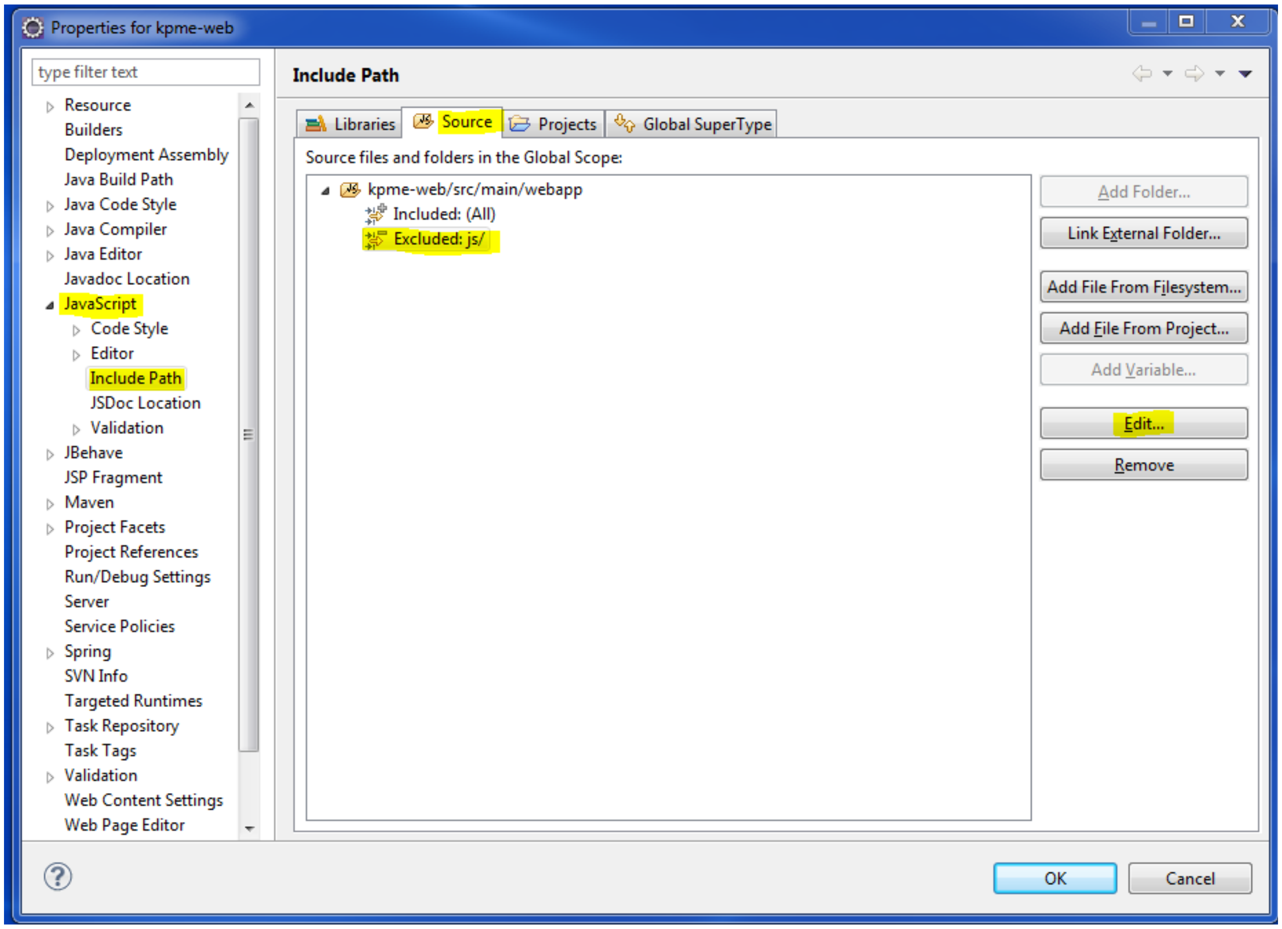
There are two methods for successfully turning off Javascript Validation in Eclipse:

1) Exclude from Path (best solution)

1.
    a. Right click your project
    b. Select Properties -> JavaScript -> Include Path
    c. Select `Source` tab. (It looks identical to Java Build Path Source tab)
    d. Expand JavaScript source folder
    e. Highlight `Excluded` pattern
    f. Click `Edit` button
    g. Click Add button next to `Exclusion patterns` box and add the exclusion for the js folder.

The information about JavaScript source inclusion/exclusion is saved into `.settings/.jsdtscope` file.

Here is how configuration looks with the js folder removed from validation

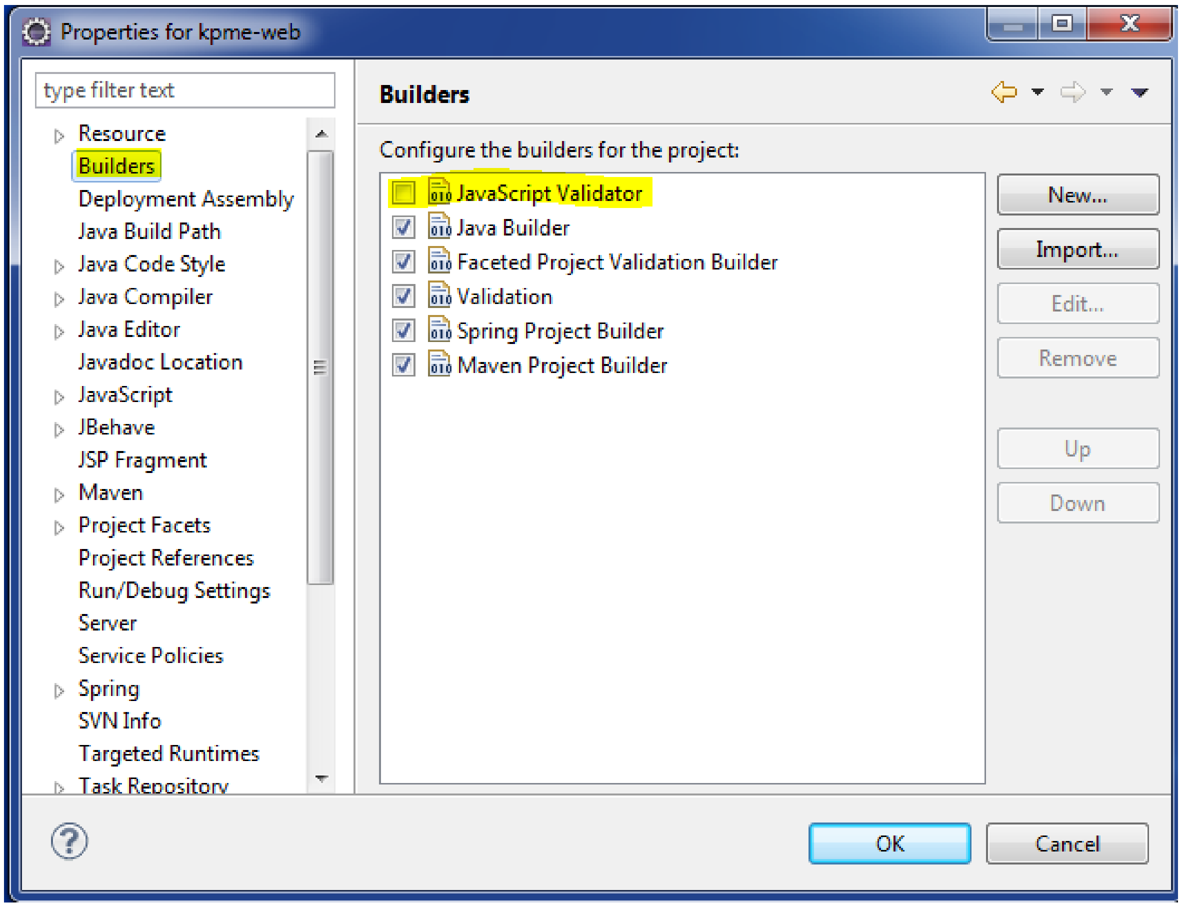


Source files and folders in the Global Scope:

kpme-web/src/main/webapp
Included: (All)
Excluded: js/

2) Turn off the JavaScript Validator in the "Builders" config for your project:

1.
    a. Right click your project
    b. Select Properties -> Builders
    c. Uncheck the "JavaScript Validator"



Configure the builders for the project:

- [ ] JavaScript Validator
- [✓] Java Builder
- [✓] Faceted Project Validation Builder
- [✓] Validation
- [✓] Spring Project Builder
- [✓] Maven Project Builder

Then restart your Eclipse.

Install Intellij

Install the Intellij IDE

- Download and install the latest "ultimate" version of Intellij for your platform at http://www.jetbrains.com/idea/download/
- Open source licenses for Kuali development are available, ask you team leader.
- Go to your install directory, then open the 'bin' folder and edit `idea64.vmoptions`. You will want to increase the base memory settings, but the below should be a good starting point.

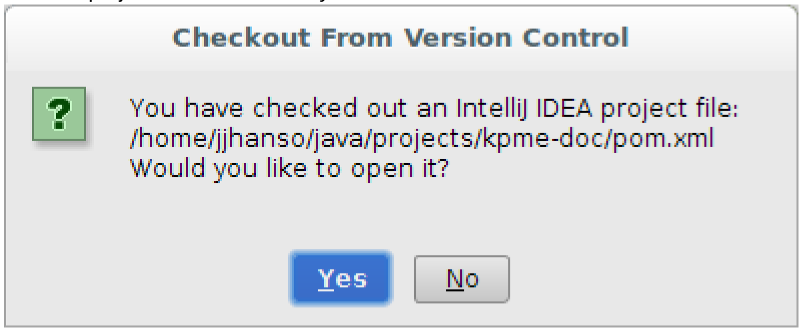
```
-Xms256m
-Xmx1024m
-XX:MaxPermSize=712m
-XX:ReservedCodeCacheSize=128m
```

- Start Intellij. It will ask you what plugins you want to enable. Selecting the default is fine.

Check out the KHR project from subversion

- Under "VCS", select "Checkout from Version Control" --> "Subversion"
- Click the green "+" button and add the kpme repository: https://svn.kuali.org/repos/kpme
- After adding the repository, it should be listed in the current window. Expand the repository and select the folder you want to check out (usually trunk or something in branches)
- Click "Checkout"
- Select your destination folder (usually something like c:/java/projects/kpme or /java/projects/kpme). You can create folders in this window if they do not exist.
  - Highlight (click) on the folder you want to check out the project to
- Click "OK"
- Click "OK" again

- Select your subversion working copy format.
    - 1.7 or 1.8 should work fine
    - generally it is a good idea to keep this in line with any other subversion client installations you are running (tortoiseSvn, collabnet, etc) to maintain compatibility
- Click "OK"
- After the project has downloaded you should see a window similar to this:



Checkout From Version Control

You have checked out an IntelliJ IDEA project file:
/home/jjhanso/java/projects/kpme-doc/pom.xml
Would you like to open it?

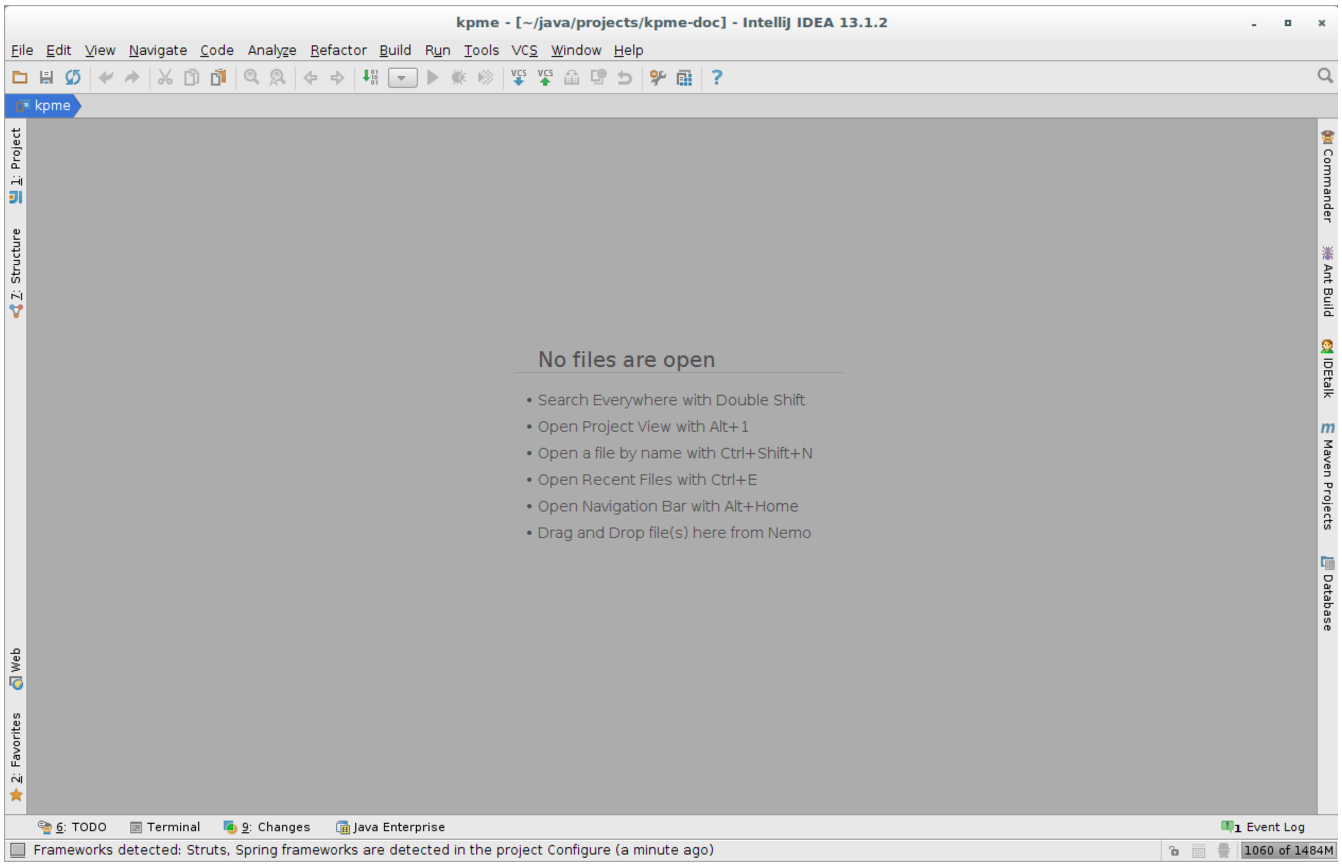
- Click "Yes"
- Intellij will then open your project, which may take several minutes.
    - During this, Intellij will ask you if you want to schedule several files for addition to subversion.
        - These are Intellij project files for each of the modules
    - Click "No"
- You may get a window saying "Language level changes will take effect on project reload. Would you like to reload the project "kpme" now?"
    - Click "Yes"
- Intellij will restart.

A few environment variables you need to set

- ANT_OPTS: -Xmx1g -XX:MaxPermSize=512m
- MAVEN_OPTS: -Xmx1g -XX:MaxPermSize=512m

A few common setup tasks

After intellij restarts, you should be presented with a gray (or black) window that looks similar to this:



No files are open

- Search Everywhere with Double Shift
- Open Project View with Alt+1
- Open a file by name with Ctrl+Shift+N
- Open Recent Files with Ctrl+E
- Open Navigation Bar with Alt+Home
- Drag and Drop file(s) here from Nemo

Frameworks detected: Struts, Spring frameworks are detected in the project Configure (a minute ago)

Couple of things to point out for new Intellij Users:

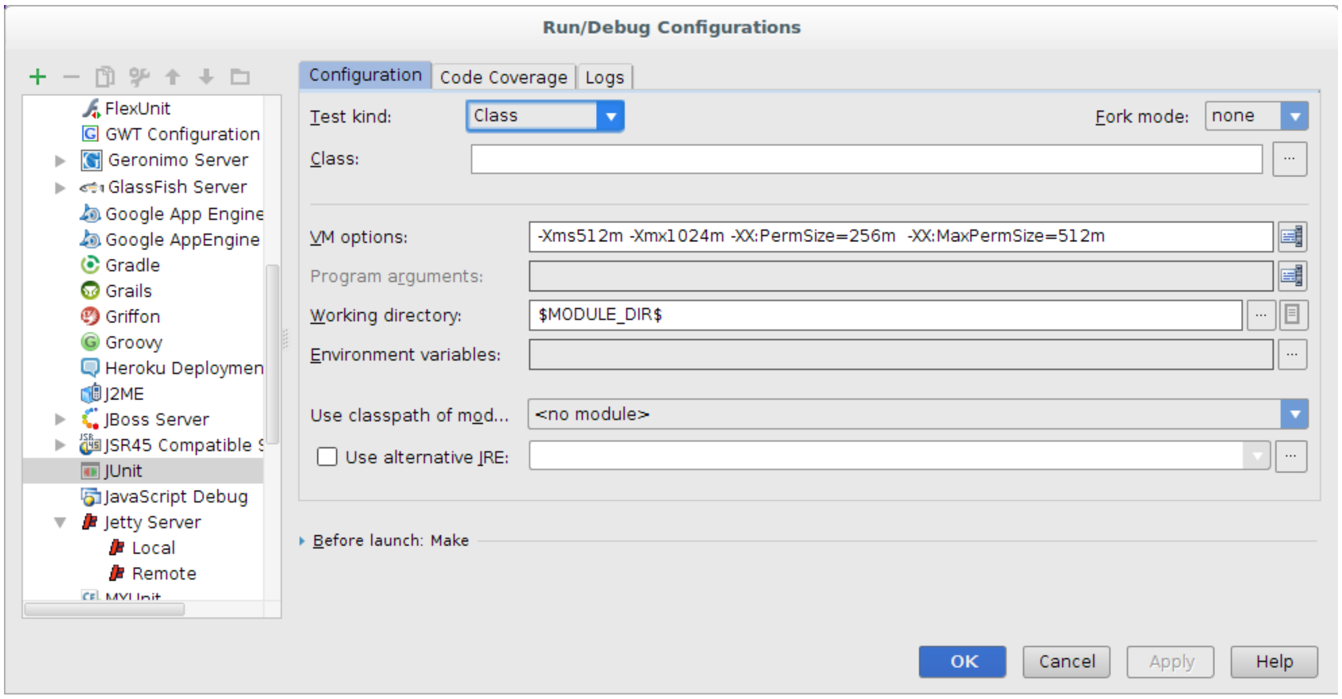
- Upper left of the workspace (big darker gray area) there is a button for "Project". Clicking this will give a very nice folder view of the project
- "Changes", bottom, next to TODO, and Terminal is your subversion client
    - Clicking this will present a window at the bottom with local changes, repository changes, and incoming changes tabs
    - To update your project to the latest code base, click Changes, Incoming, Refresh (button that looks like two arrows creating a circle), and then the Update Project button.

Setting up run configurations

Several run configurations can be set up to make life easier.

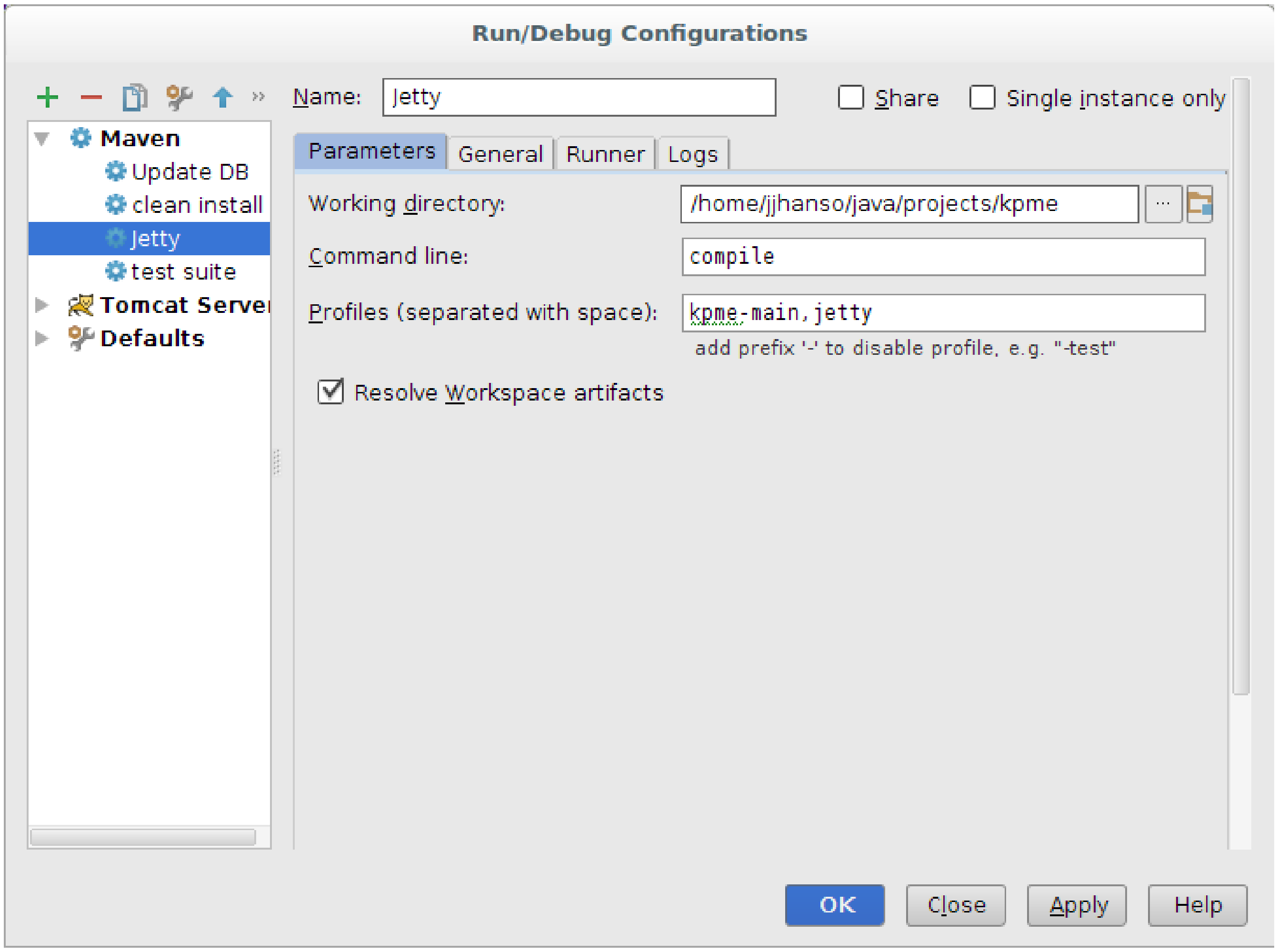
Set JUnit defaults

In the Run Configuration window (Run --> Edit Configurations), expand the Defaults, and select JUnit



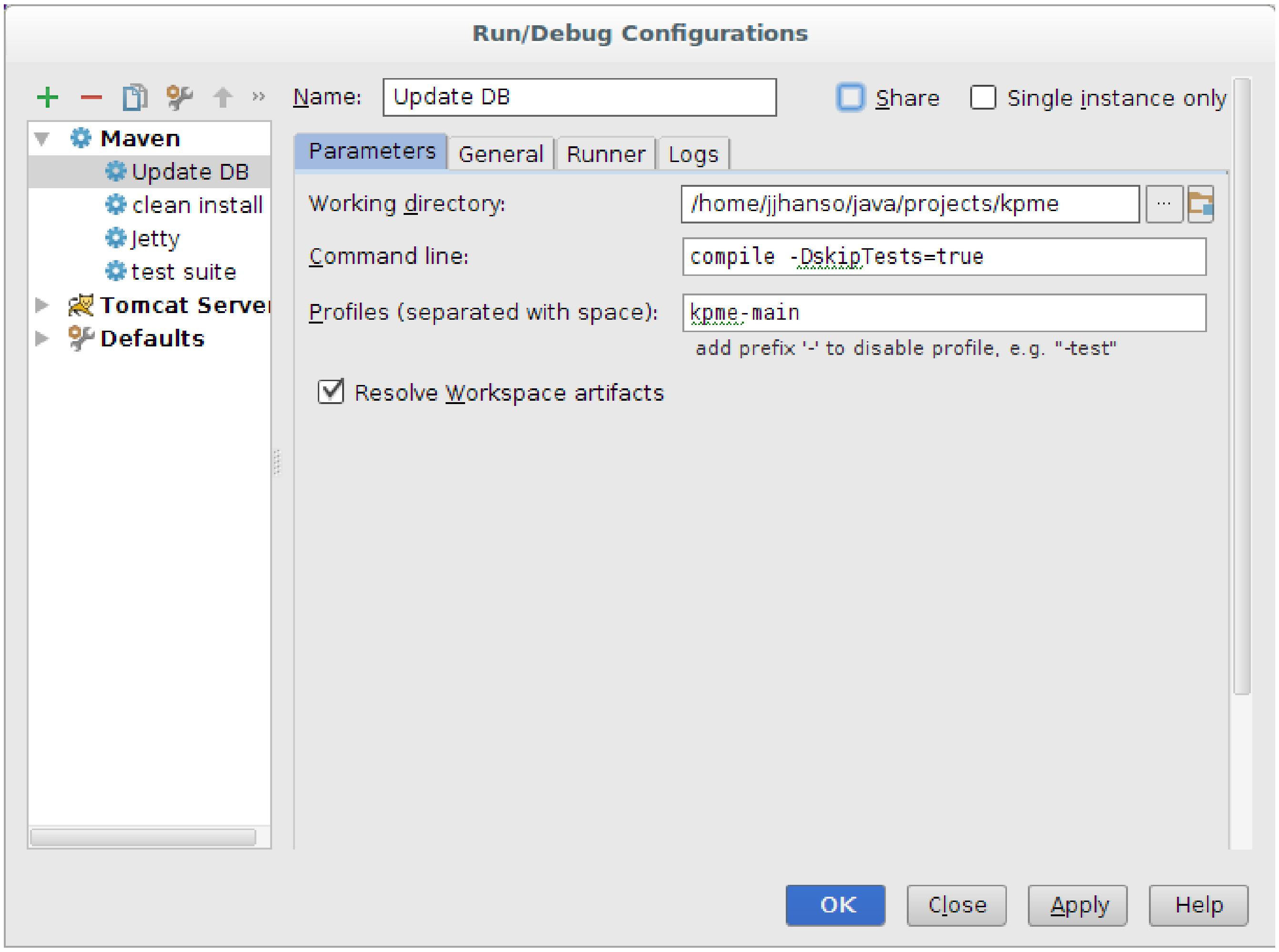
Test kind: Class
Fork mode: none
Class:
VM options: -Xms512m -Xmx1024m -XX:PermSize=256m -XX:MaxPermSize=512m
Program arguments:
Working directory: $MODULE_DIR$
Environment variables:
Use classpath of mod... <no module>
☐ Use alternative JRE:
Before launch: Make

Jetty (Using maven plugin)



Name: Jetty
☐ Share ☐ Single instance only
Working directory: /home/jjhanso/java/projects/kpme
Command line: compile
Profiles (separated with space): kpme-main,jetty
add prefix '-' to disable profile, e.g. "-test"
☑ Resolve Workspace artifacts

Update database (liquibase update)


Run/Debug Configurations

Name: Update DB

Share [ ] Single instance only [ ]

Maven
- Update DB
- clean install
- Jetty
- test suite

Tomcat Server

Defaults

Parameters | General | Runner | Logs

Working directory: /home/jjhanso/java/projects/kpme

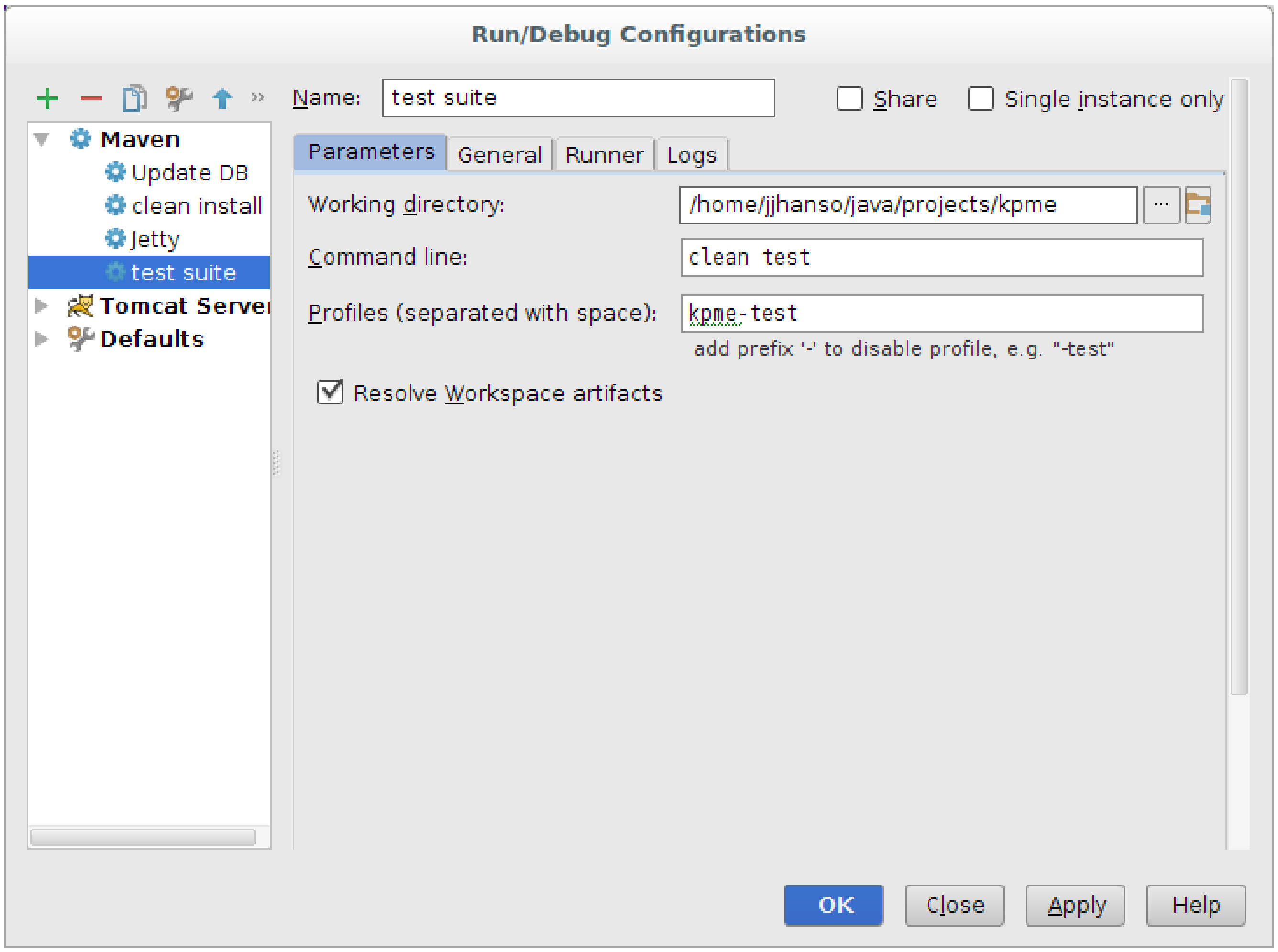
Command line: `compile -DskipTests=true`

Profiles (separated with space): `kpme-main`

add prefix '-' to disable profile, e.g. "-test"

[✓] Resolve Workspace artifacts

Run Test suite



Name: test suite

☐ Share ☐ Single instance only

Working directory: /home/jjhanso/java/projects/kpme

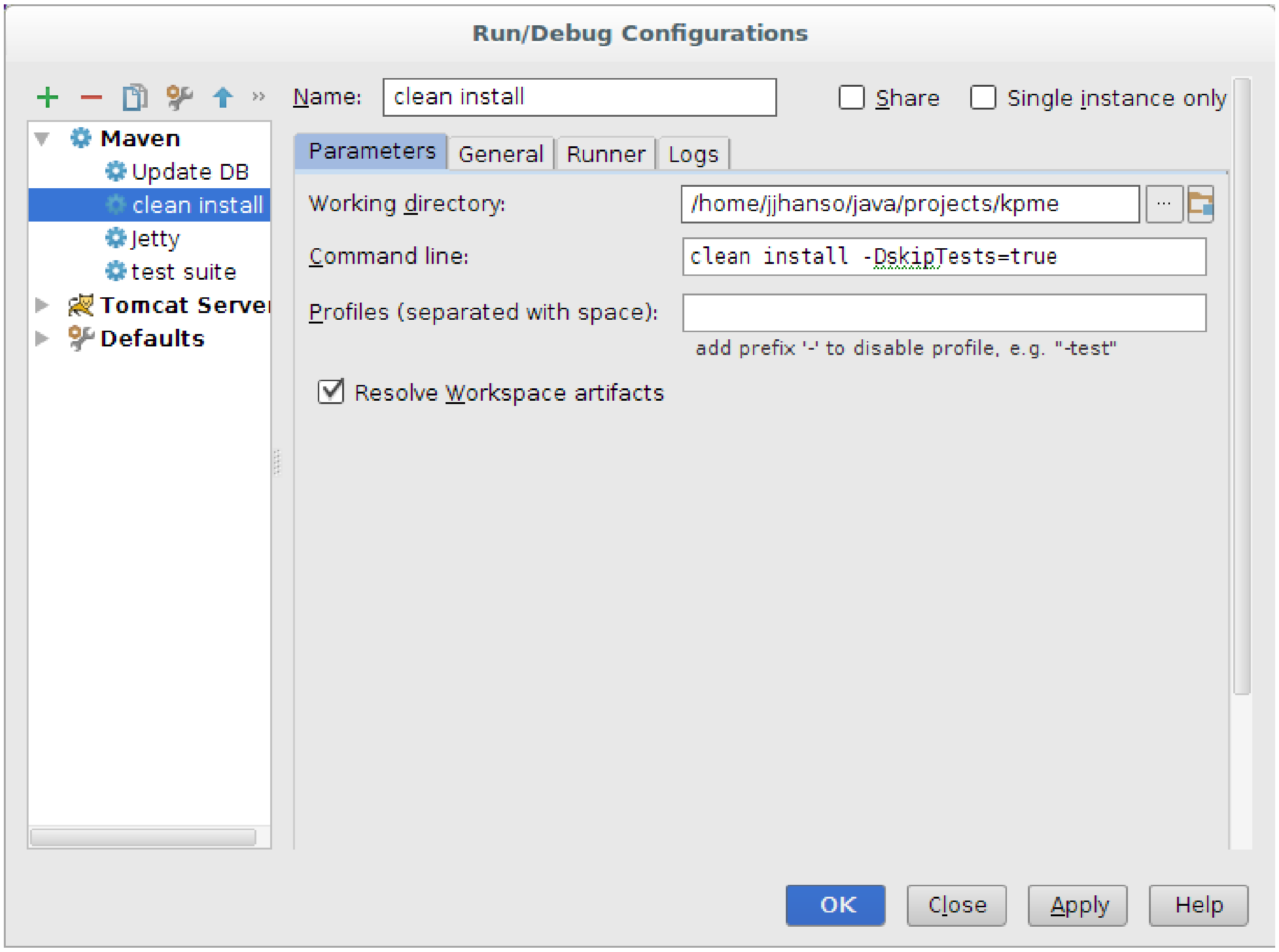
Command line: clean test

Profiles (separated with space): kpme-test

add prefix '-' to disable profile, e.g. "-test"

☑ Resolve Workspace artifacts

Maven clean and install



Name: clean install

[ ] Share [ ] Single instance only

Working directory: /home/jjhanso/java/projects/kpme

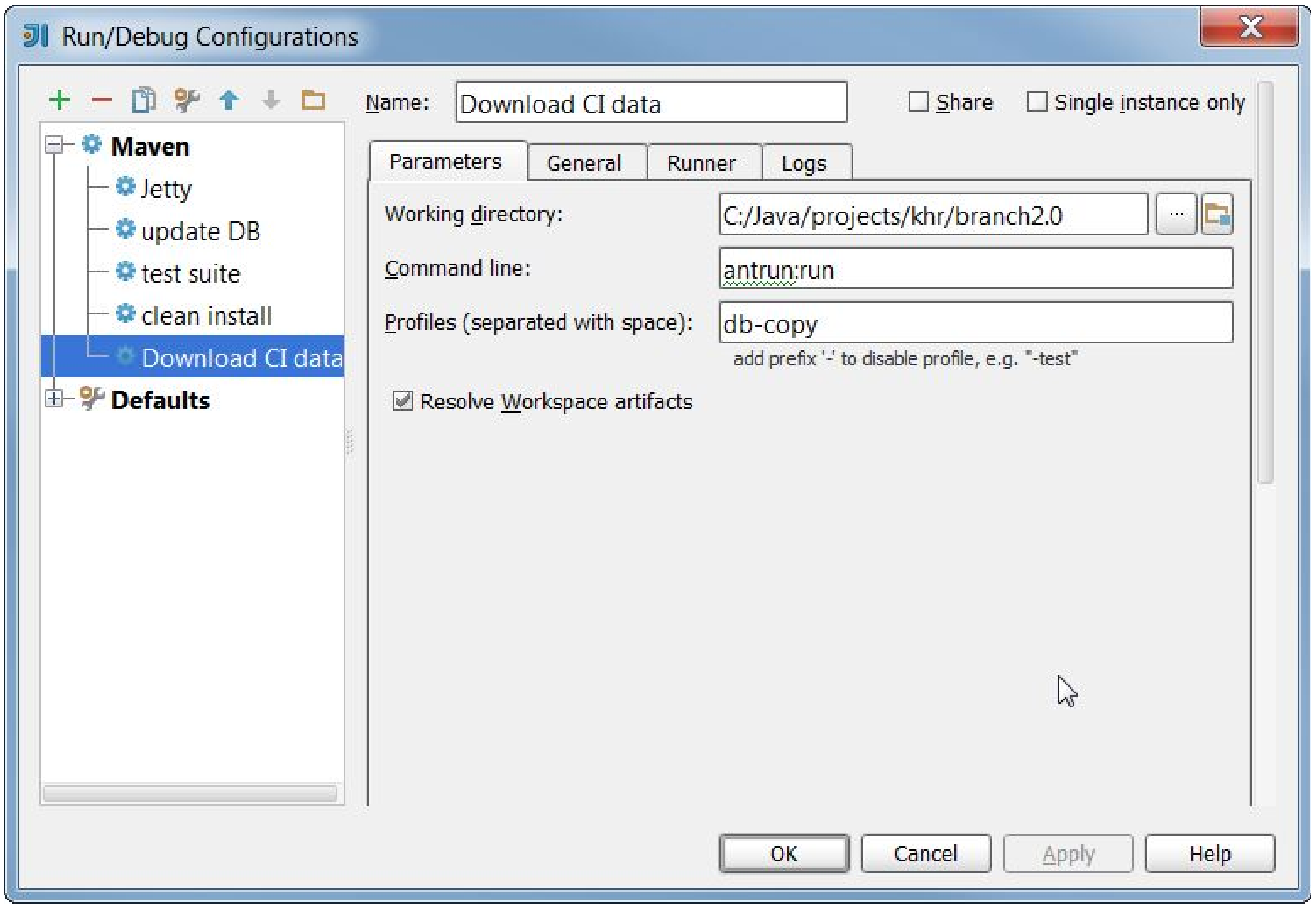
Command line: `clean install -DskipTests=true`

Profiles (separated with space):

add prefix '-' to disable profile, e.g. "-test"

[✓] Resolve Workspace artifacts

Download CI data


Name: Download CI data

☐ Share ☐ Single instance only

Maven: Jetty, update DB, test suite, clean install, Download CI data

Defaults

Parameters tab:

Working directory: C:/Java/projects/khr/branch2.0

Command line: antrun:run

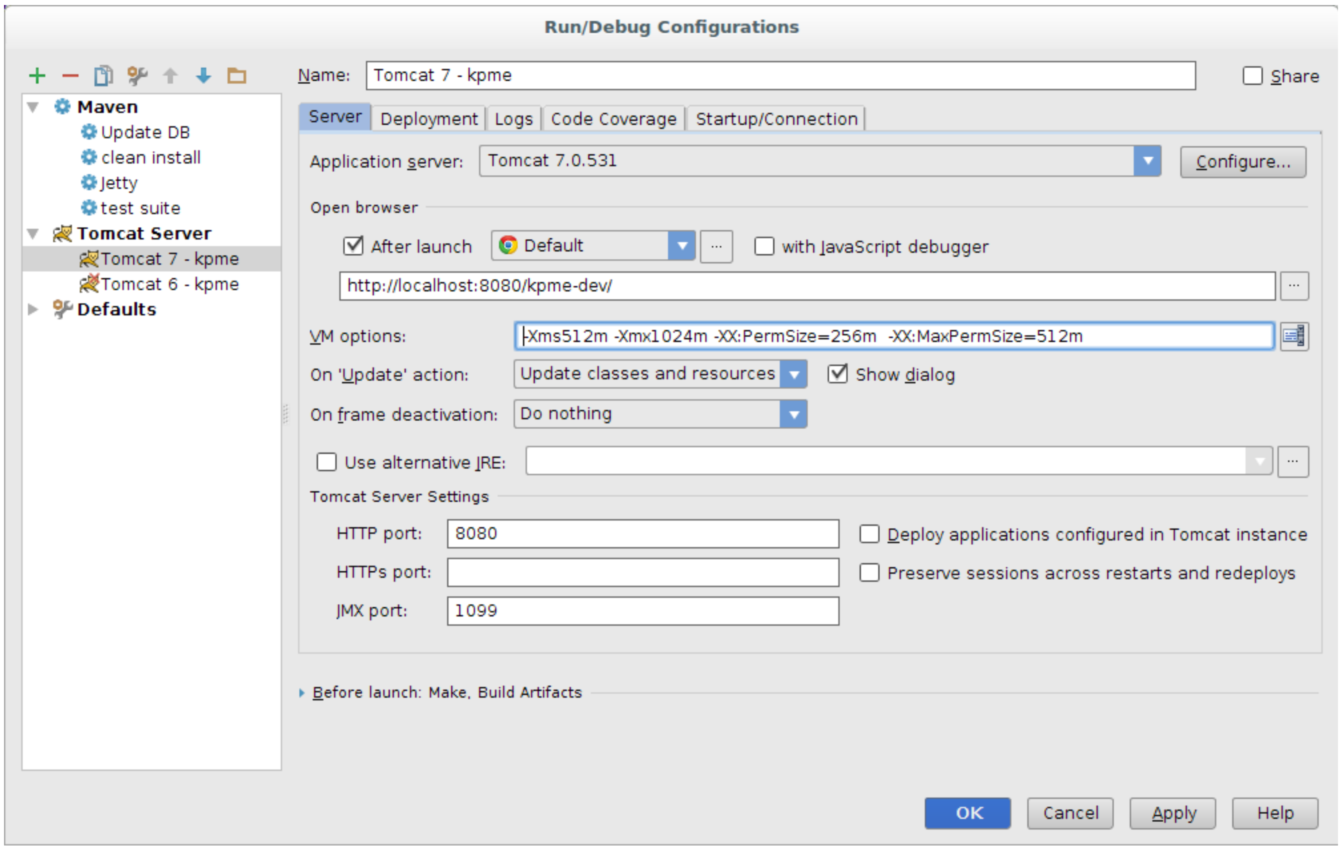
Profiles (separated with space): db-copy

add prefix '-' to disable profile, e.g. "-test"

☑ Resolve Workspace artifacts

Tomcat (next two pictures)

Note: for this to work, Tomcat needs to be installed, or unzipped on your computer somewhere. You can then tell intellij where it is located by clicking "Configure" on the screen below. Additionally, you will need to copy the jdbc driver you are using to the Tomcat Servers "lib" directory.


Run/Debug Configurations

Name: Tomcat 7 - kpme ☐ Share

Maven: Update DB, clean install, Jetty, test suite
Tomcat Server: Tomcat 7 - kpme, Tomcat 6 - kpme
Defaults

Server | Deployment | Logs | Code Coverage | Startup/Connection

Application server: Tomcat 7.0.531 Configure...

Open browser

☑ After launch Default ☐ with JavaScript debugger

http://localhost:8080/kpme-dev/

VM options: -Xms512m -Xmx1024m -XX:PermSize=256m -XX:MaxPermSize=512m

On 'Update' action: Update classes and resources ☑ Show dialog

On frame deactivation: Do nothing

☐ Use alternative JRE:

Tomcat Server Settings

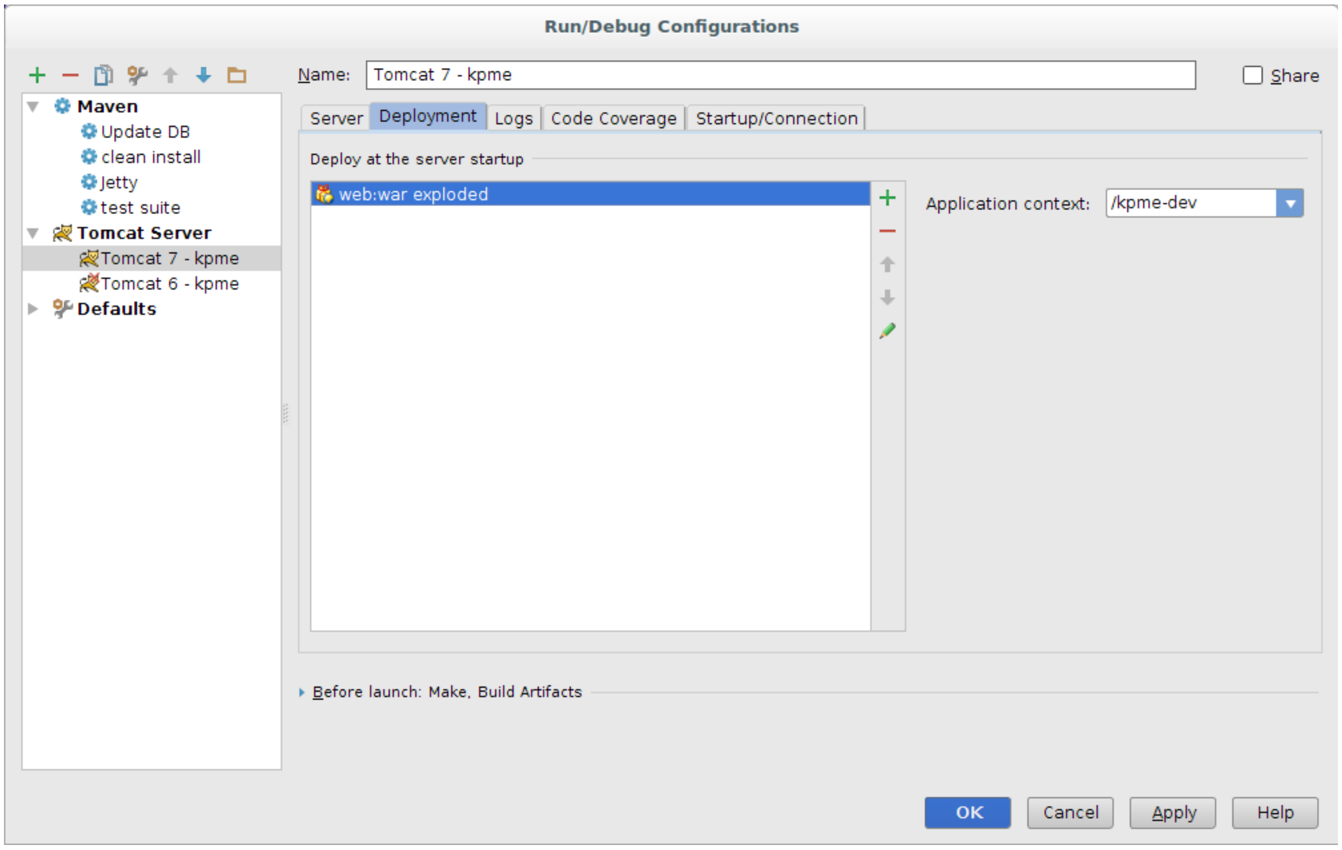
HTTP port: 8080 ☐ Deploy applications configured in Tomcat instance

HTTPs port: ☐ Preserve sessions across restarts and redeploys

JMX port: 1099

Before launch: Make, Build Artifacts

OK Cancel Apply Help


Run/Debug Configurations

Name: Tomcat 7 - kpme ☐ Share

Maven: Update DB, clean install, Jetty, test suite
Tomcat Server: Tomcat 7 - kpme, Tomcat 6 - kpme
Defaults

Server | Deployment | Logs | Code Coverage | Startup/Connection

Deploy at the server startup

web:war exploded

Application context: /kpme-dev

Before launch: Make, Build Artifacts

OK Cancel Apply Help