

Errai Basic setup

Author:	Sebastian Börebäck
Version	1.0 Initial draft
Date	2013-04-04

Contents

Why	1
Purpose.....	1
Audience.....	2
My Background.....	2
Requirements	2
Important for your Eclipse settings.....	3
First to figure out which Archetype is up to date!	3
Maven installation	4
Using Eclipse.....	4
Using command line.....	8
How to run your new project.....	12
Extra	17
Extract Jboss package.....	18
Add to your Jboss server.....	19

Why

If you had problems with the quick guide or the demos from the Errai website,
This is my tips to get you started.

Purpose

This tutorial will explain how to use maven to deploy your Errai project.
If you're not familiar with maven and Errai, this is a good starting point to get some more knowledge.

I recommend to first reading the quick start guide and other info at:
<https://docs.jboss.org/author/display/ERRAI/Quickstart+Guide>

Warning When I read this and tried to follow their tutorial I failed. And I tried for days to get it working. When I was trying, the Errai team managed to release 2 new version 2.2.0 final and 2.3.1 RC1 (release candidate 1).

Audience

You already understand Java and used eclipse.
And now you really want to try out Errai.

If you missed to watch the Errai videos, here are the links:

http://www.youtube.com/watch?feature=player_embedded&v=L8davgTzme9

http://www.youtube.com/watch?v=-kuCbRDVf3Q&feature=player_embedded

http://www.youtube.com/watch?v=r3sbgCfP4AE&feature=player_embedded

Or go to the [Errai blog](#).

You have some basic understanding of Maven or at least heard about it.

My Background

My background comes from working as a frontend gwt developer. I found Errai because of their messengerbus and directly wanted to start using it. Then I noticed it has CDI, REST etc. Lots of candy and bling!

I have some experience with Maven and Gwt and Guice (which is Google's version of CDI.). So I understand the concepts. I understand good GWT, but the CDI pattern/framework is still complex. And Maven is for me like windows printers/drivers a black box which make things work sometimes but if you poke it too much it breaks.

Requirements

So you want to try out Errai.

First to create an Errai project you need these tools:

- Eclipse
- Maven plugin
- Gwt package
- Errai package
- Java 6

I'm going to use for this tutorial:

- Eclipse Java EE IDE for Web Developers. Juno
- Gwt 2.4
- Errai 2.3.0.CR1 (2.2 final works as well)
- Java 1.6_0_35 sdk
- Maven 3.0
- Windows 7

Extra what I have installed in eclipse is and is not needed for the project to run:

- Aptana studio 3 - useful for web design
- Jboss developer studio 6 (which has all the useful tools for Errai and Jboss)
- Gwt Designer
- Google app engine

Important for your Eclipse settings.

The project need to be using **JAVA 1.6** and currently I recommend using **GWT 2.4** (2013-04-02) . The newer Errai versions will support 2.5 and this example project may work with 2.5.

Some basic things that you need to have setup before you start:

In you path you need to make sure that `java_home` is pointing to java 1.6 sdk.

In terminal/cmd when you type `echo %java_home%` you get the path to `C:\...\java\jdk1.6..`

And that you maven path is also setup check by `echo %m2_home%`
`C:\...\apache-maven-3...`

To "fast" get started with Errai projects the Errai team offer **Maven Archetypes** . Which I think is cool but since the Errai projects update allot (this is awesome because you know is something growing) the guide breaks allot which sucks. The Maven Archetypes are project skeletons, which mean that when you download one of them you have a good start point for creating your webapp. It adds all the basic classes to your project.

Jboss also offers something called Forge. Which I Lincoln Baxter III talks about in this [video](#).

This is even cooler than Maven Archetypes.(Not covered in this tutorial)

First to figure out which Archetype is up to date!

Check out the <http://search.maven.org/> and search for `org.jboss.errai.archetypes`

Which are the Archetypes available for building maven to create Errai projects.

<http://search.maven.org/#search%7Cga%7C1%7Corg.jboss.errai.archetypes>

Search Results

GroupId	ArtifactId	Latest Version
org.jboss.errai.archetypes	archetypes-parent	2.3.0.CR1 all (18)
org.jboss.errai.archetypes	jaxrs-quickstart	2.3.0.CR1 all (18)
org.jboss.errai.archetypes	cdi-quickstart	2.3.0.CR1 all (18)
org.jboss.errai.archetypes	bus-quickstart	2.3.0.CR1 all (18)
org.jboss.errai.archetypes	jboss-errai-kitchensink-archetype	2.2.1.Final all (10)
org.jboss.errai.archetypes	kitchensink-quickstart	2.0.2.Final all (8)

Screen clipping taken: 2013-04-02 15:44

Here you see The **ArtifactId** and more important **Latest Version**.

From this information you can see that jaxrs-quickstart, cdi-quickstart, bus-quickstart are all using 2.3.0.CR1 which is the newest at this time Errai (up to date).

Maven installation

[Wiki](#)

Is a fast way to create project. With help of a pom.xml file you can fast configure I need these jars I'm my project and then boom you have a project with those jars. Updating the jars to newer version is just changing a variable. With that said is also very hard to get a good overview if you not used to Maven.

For this tutorial I'm going to use the cdi-quickstart archetype.
(which is as you see up to date).

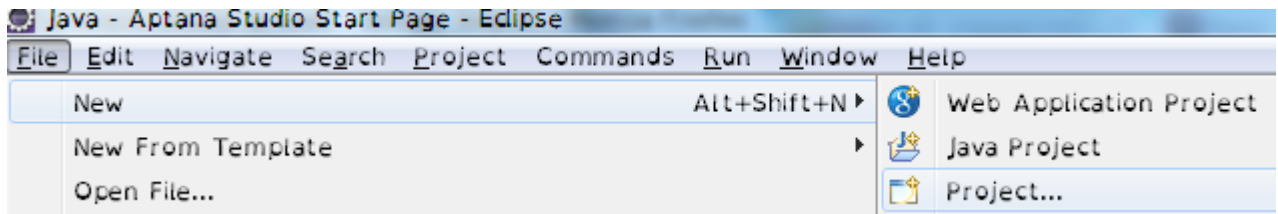
We need to download the archetype using maven and then in eclipse import it.

Because you wont see the "skeleton" before its downloaded and imported in eclipse.

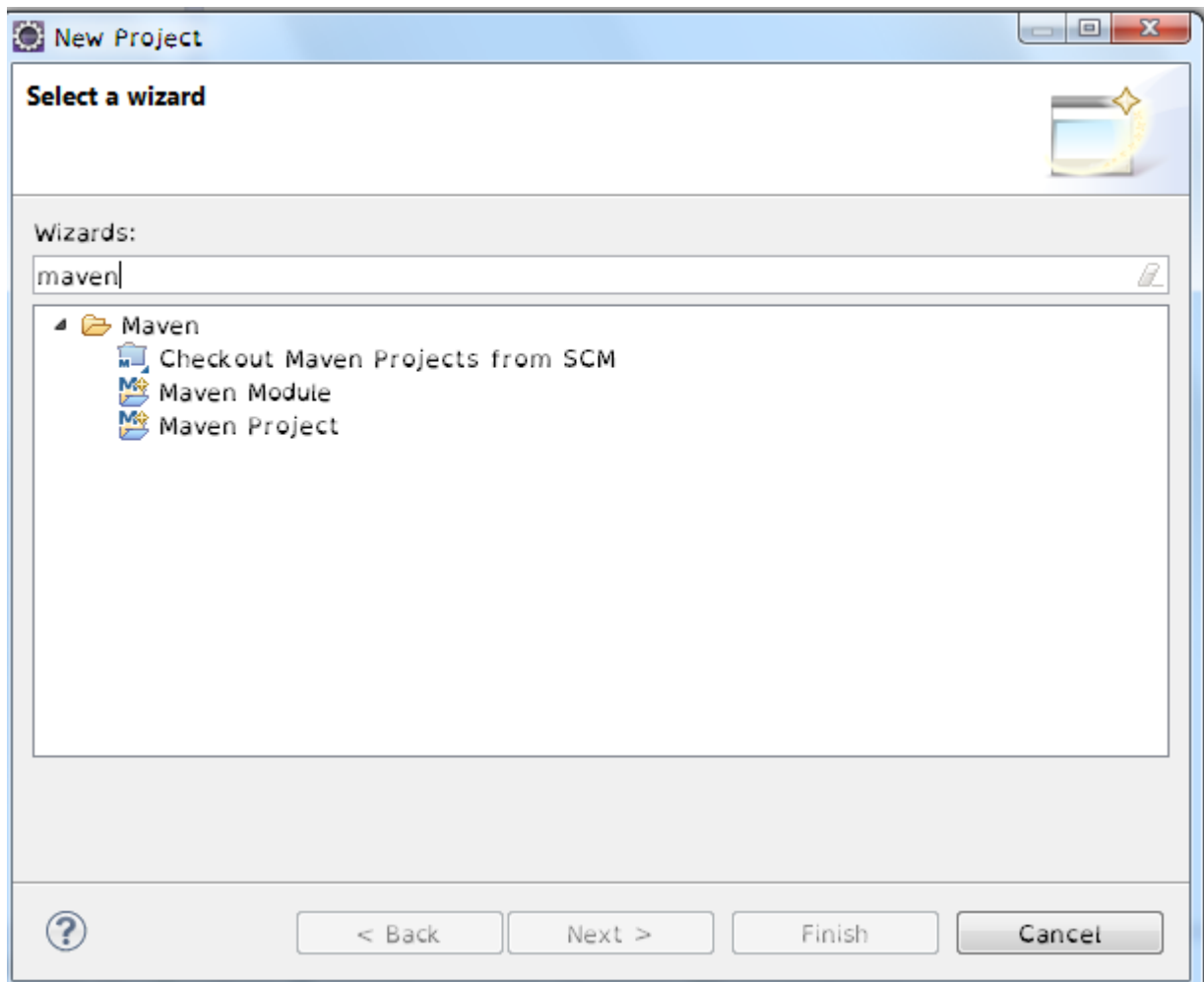
There is 2 ways to do this. Either command line with mvn or using eclipse (you need to have maven plugin installed and [the catalogs](#)).

Using Eclipse

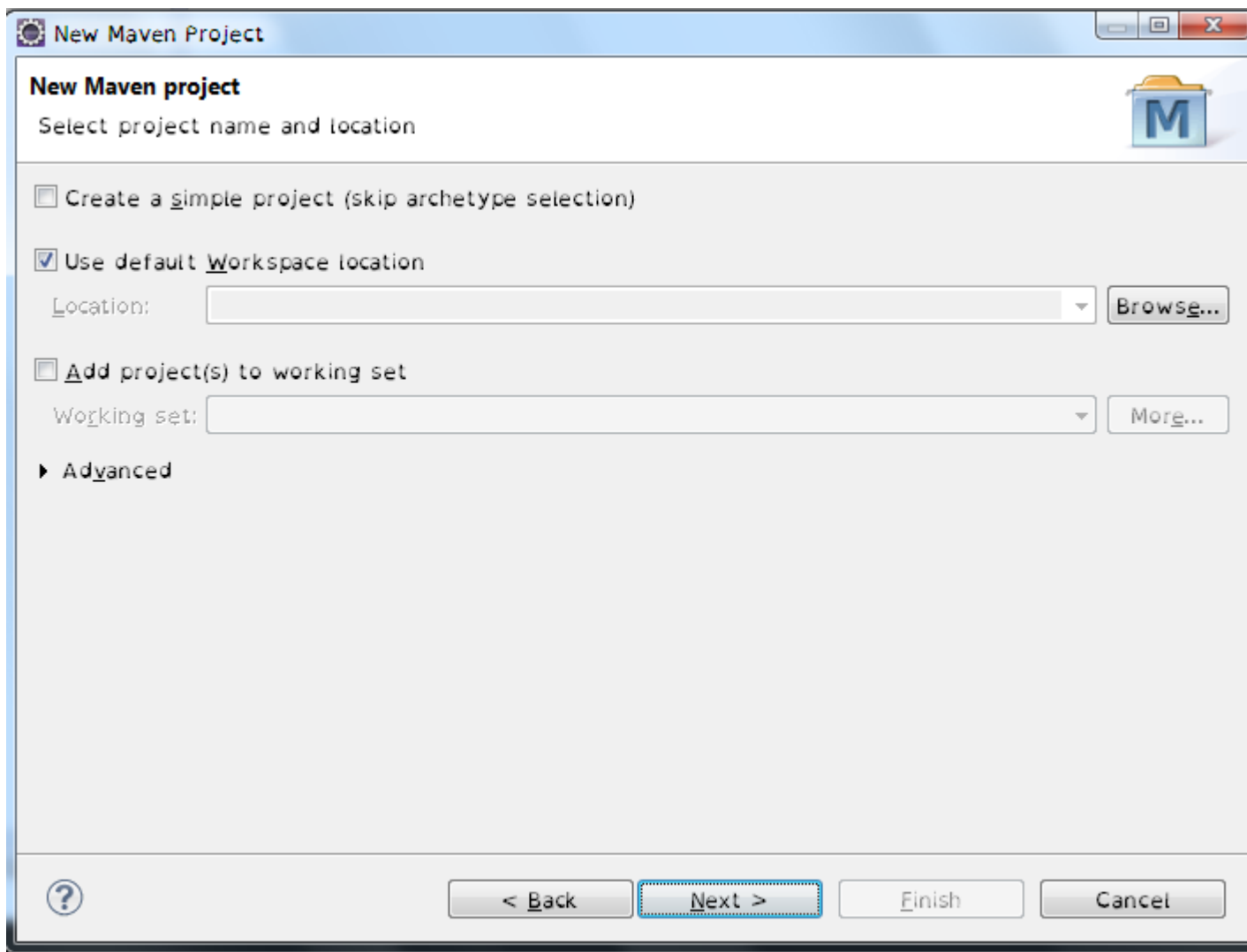
In eclipse select File --> New --> project....



Type maven and select Maven Project --> click next

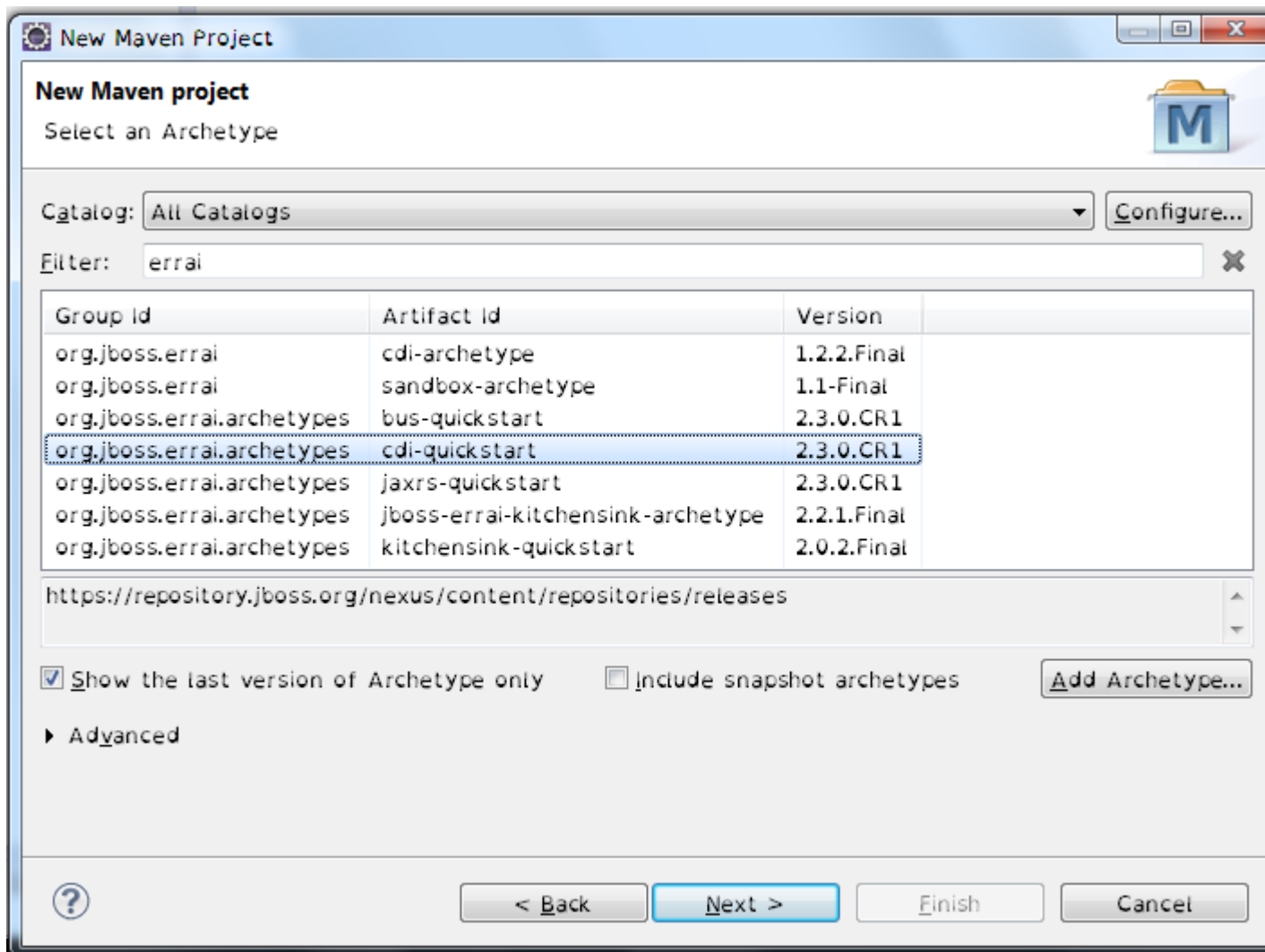


Click next



Select Catalog Nexus Indexer alternative All Catalogs. Wait a while for the Archetypes to get downloaded/updated.

- In filter type : Errai
- Select cdi-quickstart
- Click next



Now you need to fill in some info about your new Errai project:

See maven naming convention for [more info](#)

Basically:

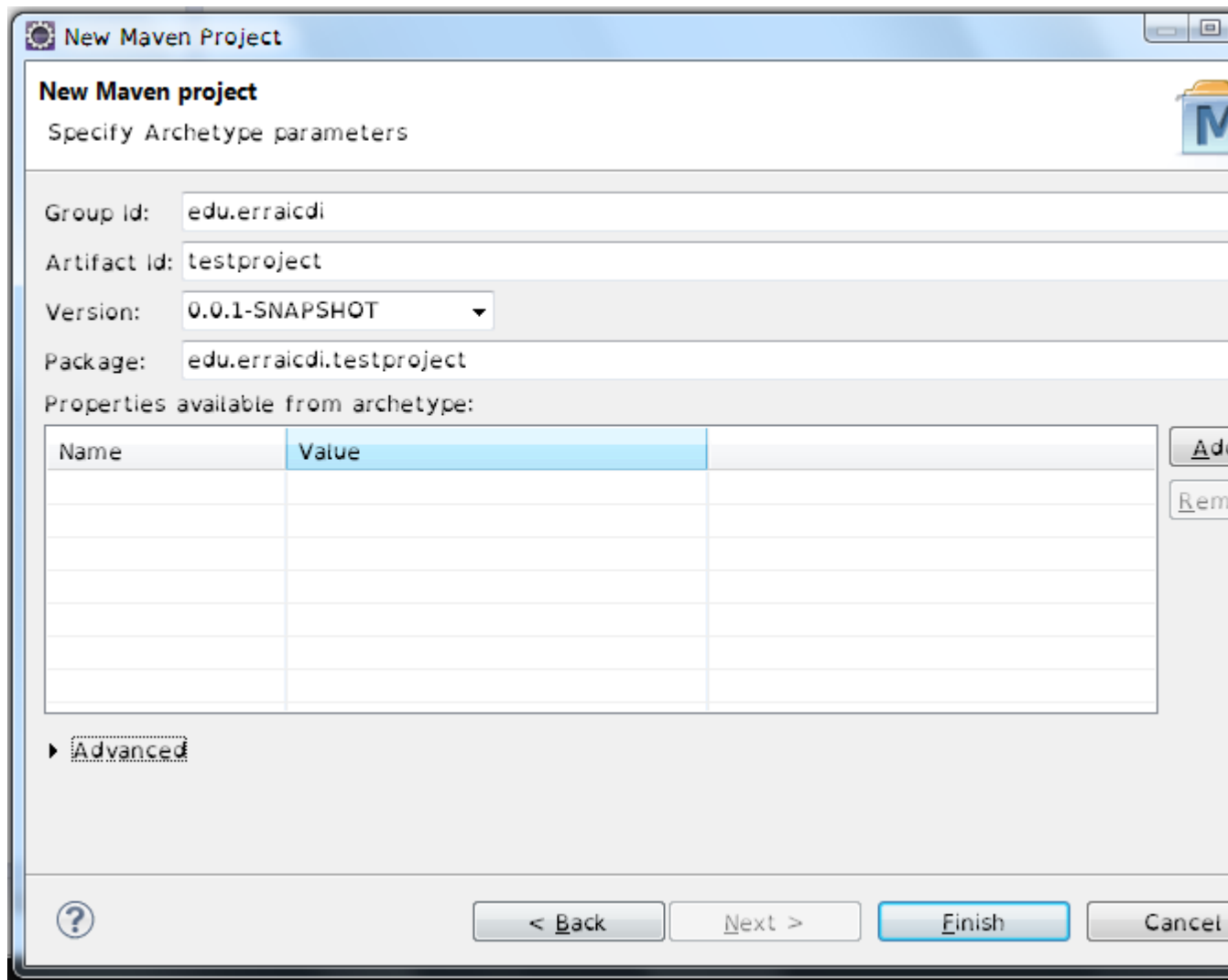
- groupId = this is package. The path to your project.
- artifactId is the name of the jar without version = the project name
- Version = is the starting version.

Here is my case: package/groupId is edu.errai:cdi (education and errai:cdi)

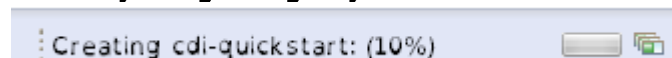
Projectname/artifactId = testproject

Version= 0.0.1 first version

When done click Finish



If everything is right you will see in the lower right corner of eclipse:



And soon a project popping up.

Done for creating the Archetype skeleton using Eclipse.

Using command line

Open cmd/terminal and go to a path where you want to download the skeleton/archetype. You need later to import this into your workspace. (good place you workspace folder).

Type/add in one line:

```
mvn archetype:generate -DarchetypeGroupId=org.jboss.errai.archetypes -
DarchetypeArtifactId=cdi-quickstart -DarchetypeVersion=2.3.0.CR1
```


So what is this:

- archetype:generate = create from archetype
- DarchetypeGroupId=org.jboss.errai.archetypes = from package we want to get the archetype. Here from the org.jboss.errai.archetypes.
- DarchetypeArtifactId=cdi-quickstart = which project we want to base our project on. Here cdi-quickstart.
- DarchetypeVersion=2.3.0.CR1 = which version of the project we want to use. Meaning which Errai version. Here 2.3.0.CR1.

```
D:\programming\Tutorial_Workspace\test2project>mvn archetype:generate
-DarchetypeArtifactId=cdi-quickstart -DarchetypeVersion=2.3.0.CR1
[INFO] Scanning for projects...
[INFO]
[INFO] -----
[INFO] Building Maven Stub Project (No POM) 1
[INFO] -----
[INFO]
[INFO] >>> maven-archetype-plugin:2.2:generate (default-cli)
[INFO] <<< maven-archetype-plugin:2.2:generate (default-cli)
[INFO] --- maven-archetype-plugin:2.2:generate (default-cli)
[INFO] Generating project in Interactive mode
```

You will then be asked to add your own groupId, artifactId and version, package. See maven naming convention for [more info](#)

Basically:

- groupId = this is package. The path to your project.
- artifactId is the name of the jar without version = the project name
- Version = is the starting version. DONT NEED TO PROVIDE (Default is 1.0)
- Package = which package. DONT NEED TO PROVIDE (Default is the groupId)

Here is my case: package/groupId is edu.erraitesting (education and erraitesting)

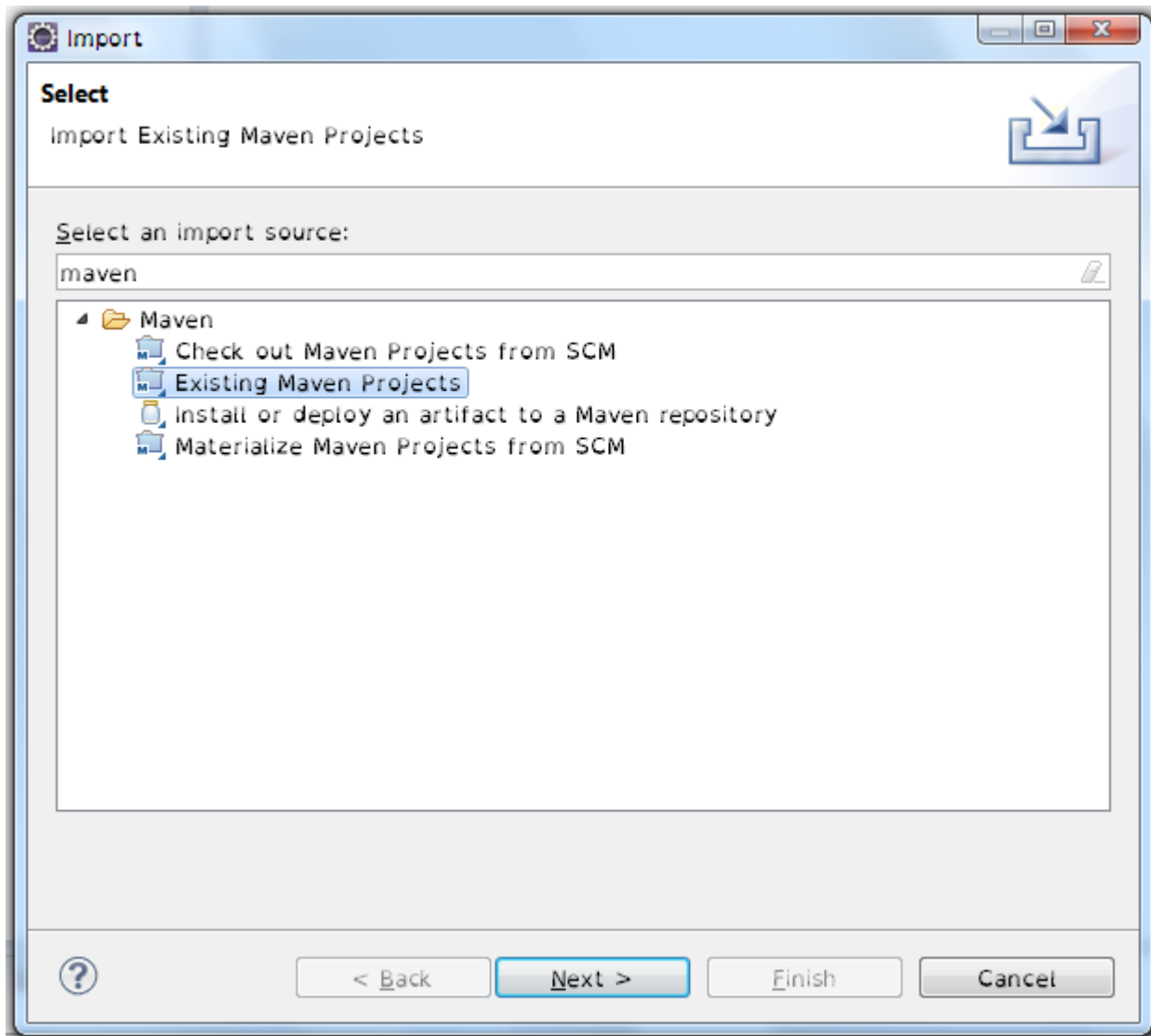
Projectname/artifactId = testErrai

Version= 1.0-SNAPSHOT first version

Answer Y to confirm everything and create the project.

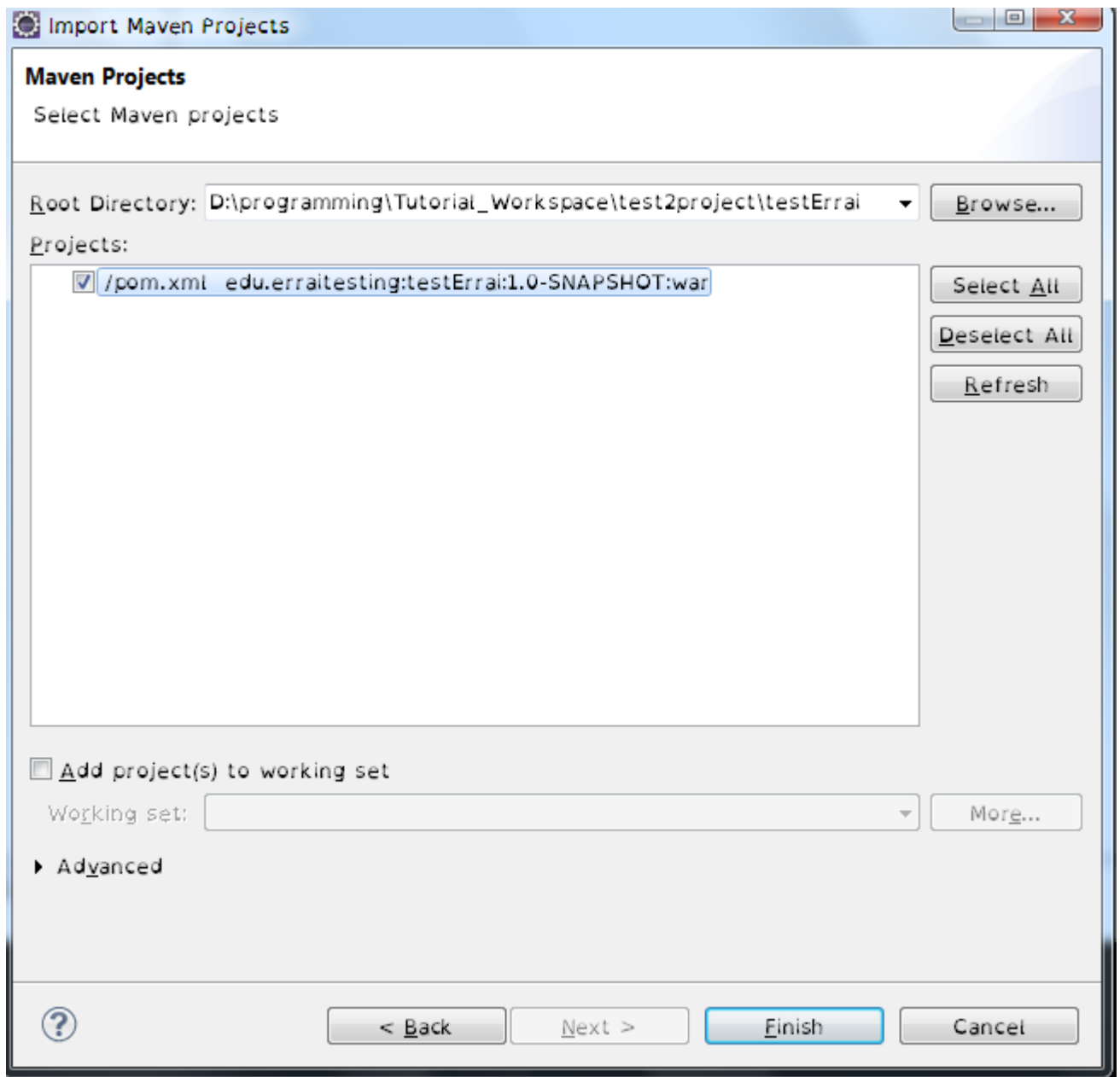
```
] found in catalog remote
Define value for property 'groupId': : edu.erraitesting
Define value for property 'artifactId': : testErrai
Define value for property 'version': 1.0-SNAPSHOT: :
Define value for property 'package': edu.erraitesting: :
Confirm properties configuration:
groupId: edu.erraitesting
artifactId: testErrai
version: 1.0-SNAPSHOT
package: edu.erraitesting
Y: : y
```

Now start Eclipse and import the project.
File --> import..
Type maven and select Existing Maven Projects
Click next

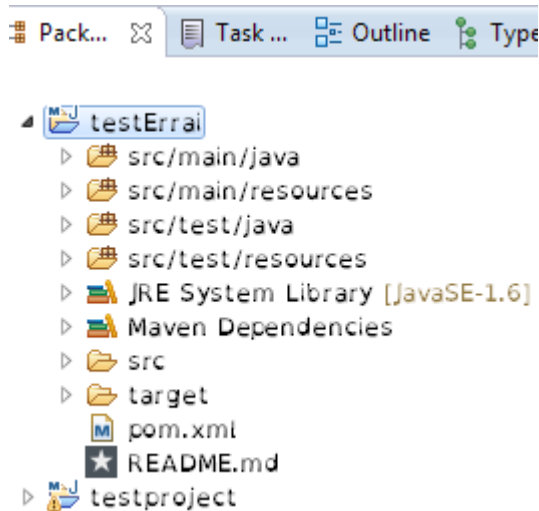


In the Root Directory: provide the path for your newly created project.
Either by copy pasting or browsing.
Wait a second. When everything worked you should see /pom.xml

My (windows) path where:
D:\programming\Tutorial_Workspace\test2project\testErrai
Then click Finish



And there should popup an new project in your eclipse



How to run your new project

In cmd/Terminal:

To run a maven gwt project just type `mvn gwt:run` in the console.

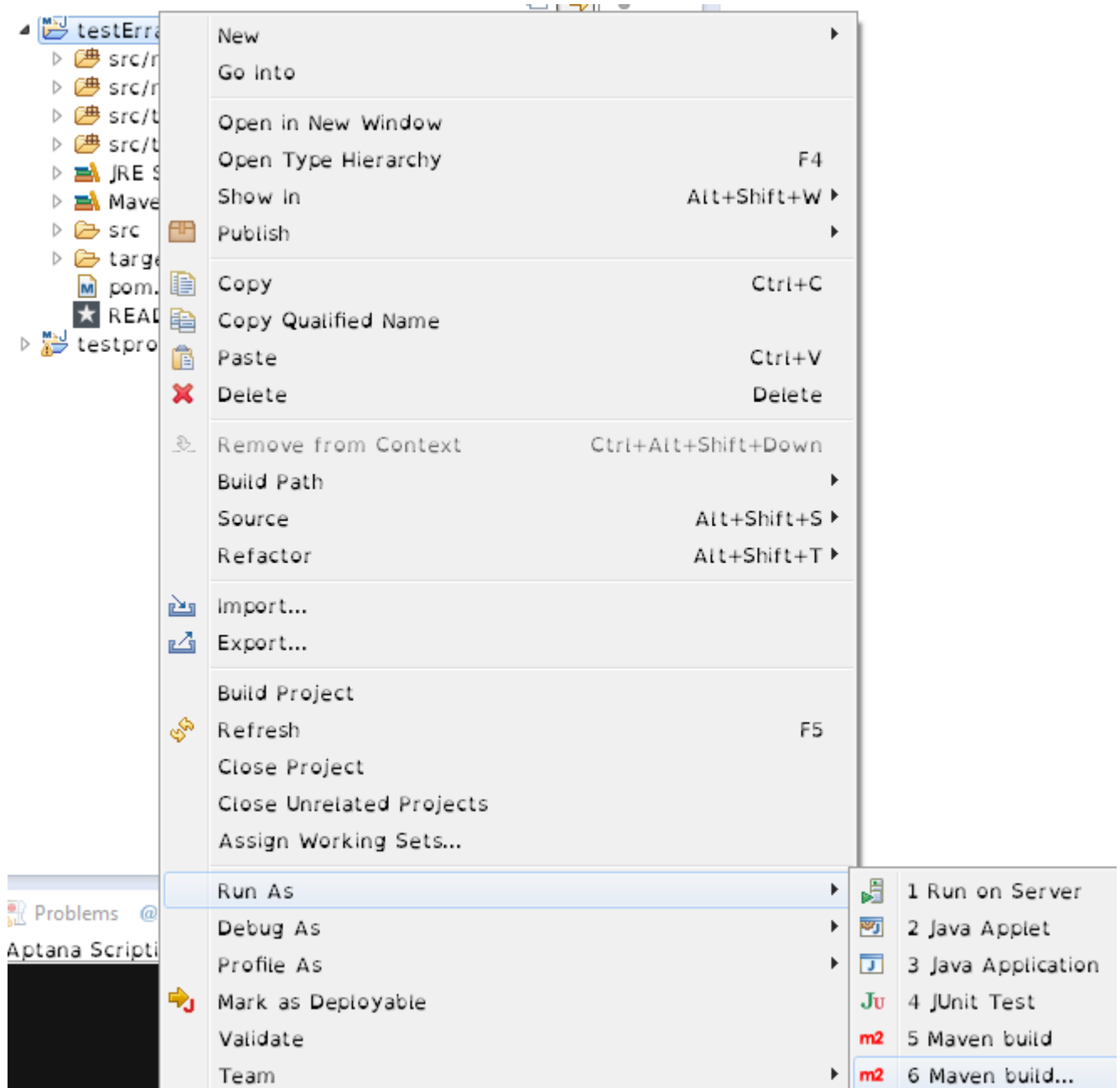
To build the maven project : `mvn install`

Better build is to clean the build folder first: `mvn clean install`

The build will then create a folder called `target` in your project folder where all the class war stuff are.

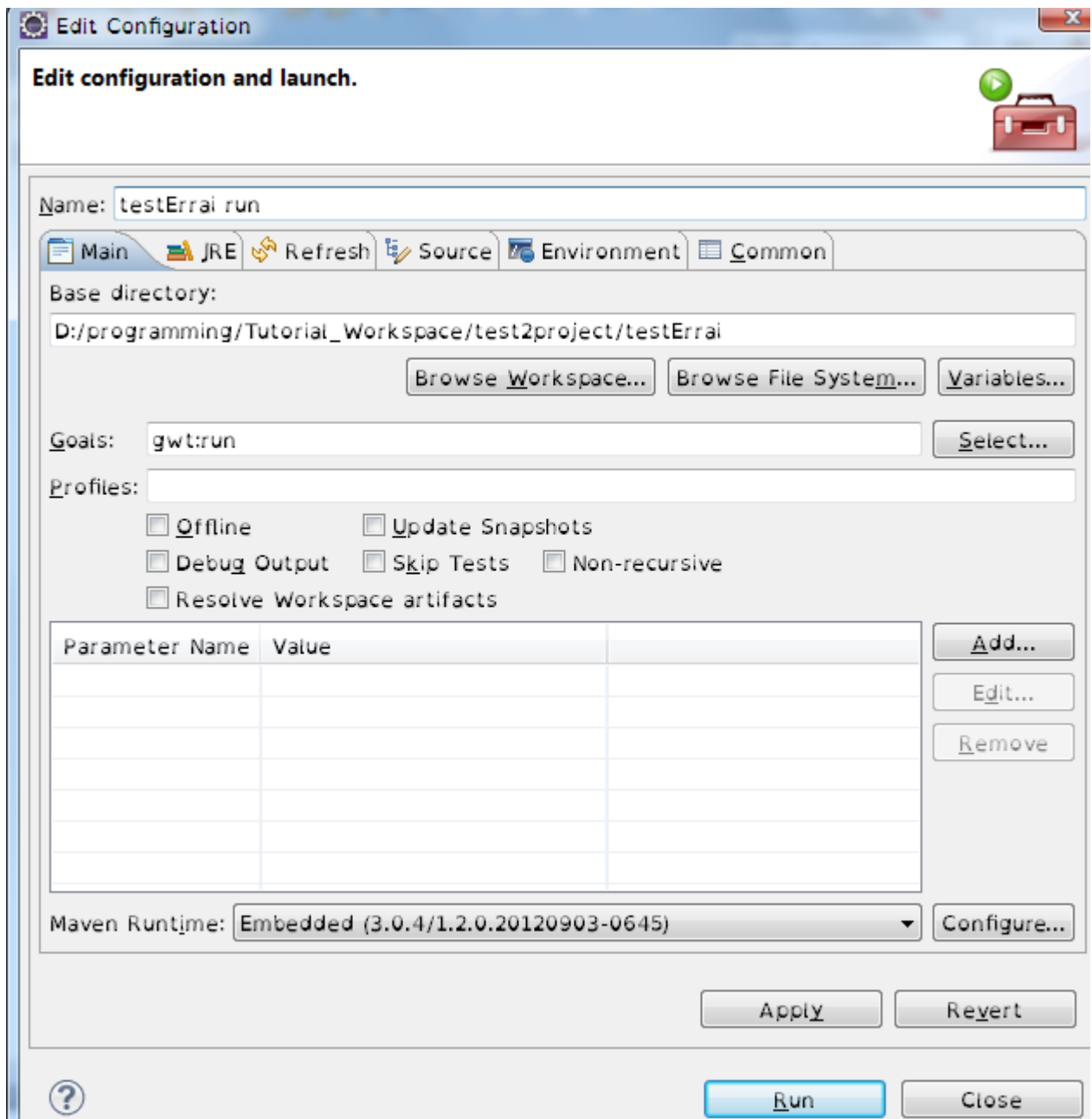
In eclipse:

Right click your project and select Run as --> **6 Maven Build...**



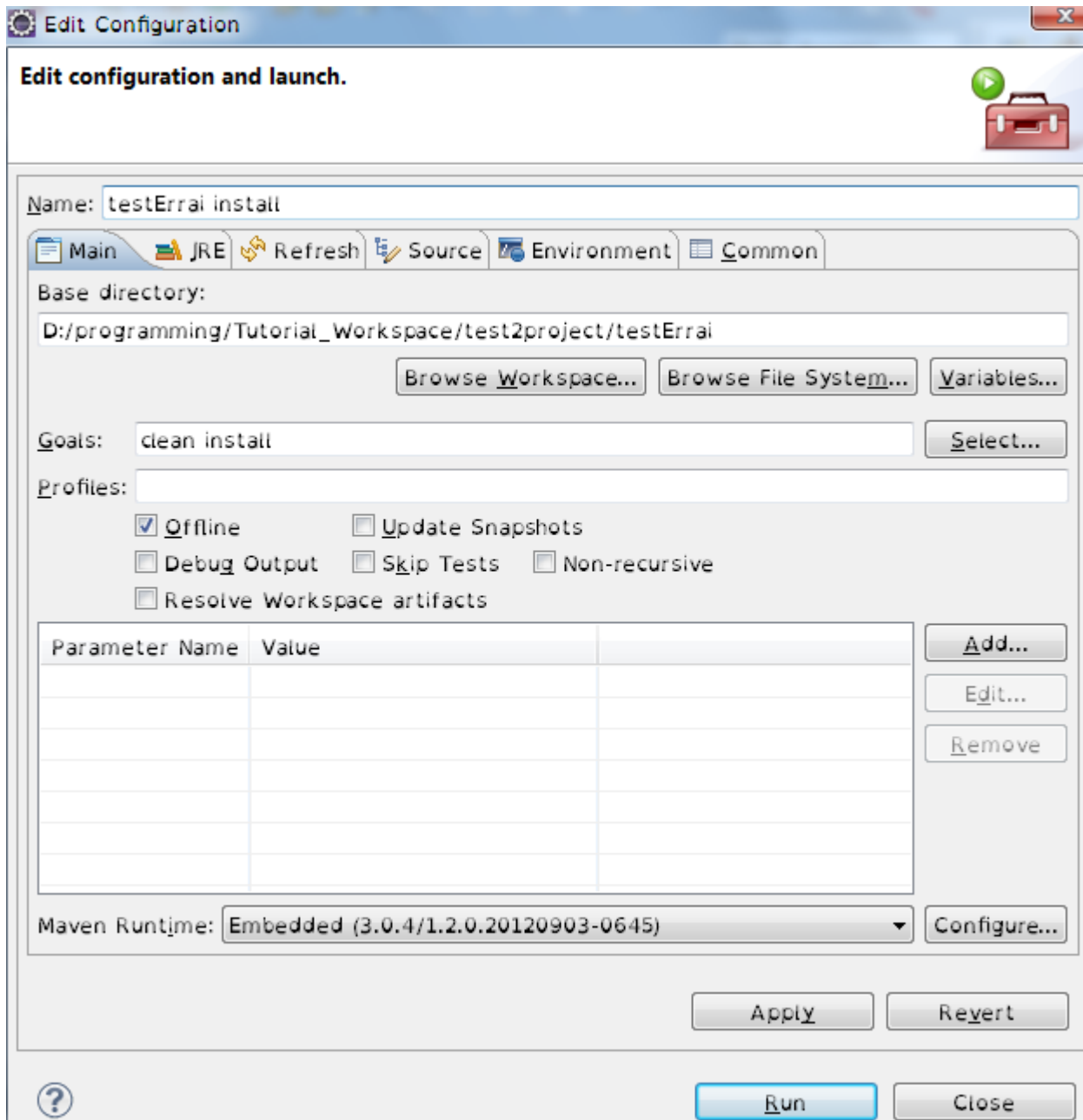
Here in the Goals: textbox type for,
 Running the gwt project= gwt:run
 To build the maven project: install
 Better build is to clean the build folder first: clean install

In Name: textbox here you will set the name of your maven run as. Which
 you later can reuse. Good to name it after what it does like: testErrai run
 first project name and then what to do= run
 Click apply (to save the config) and Run.

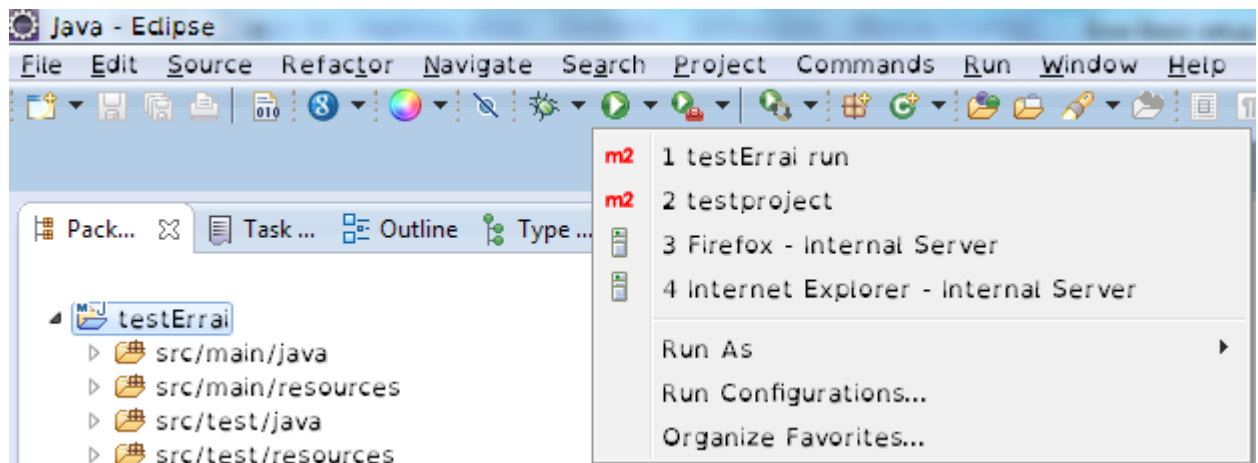


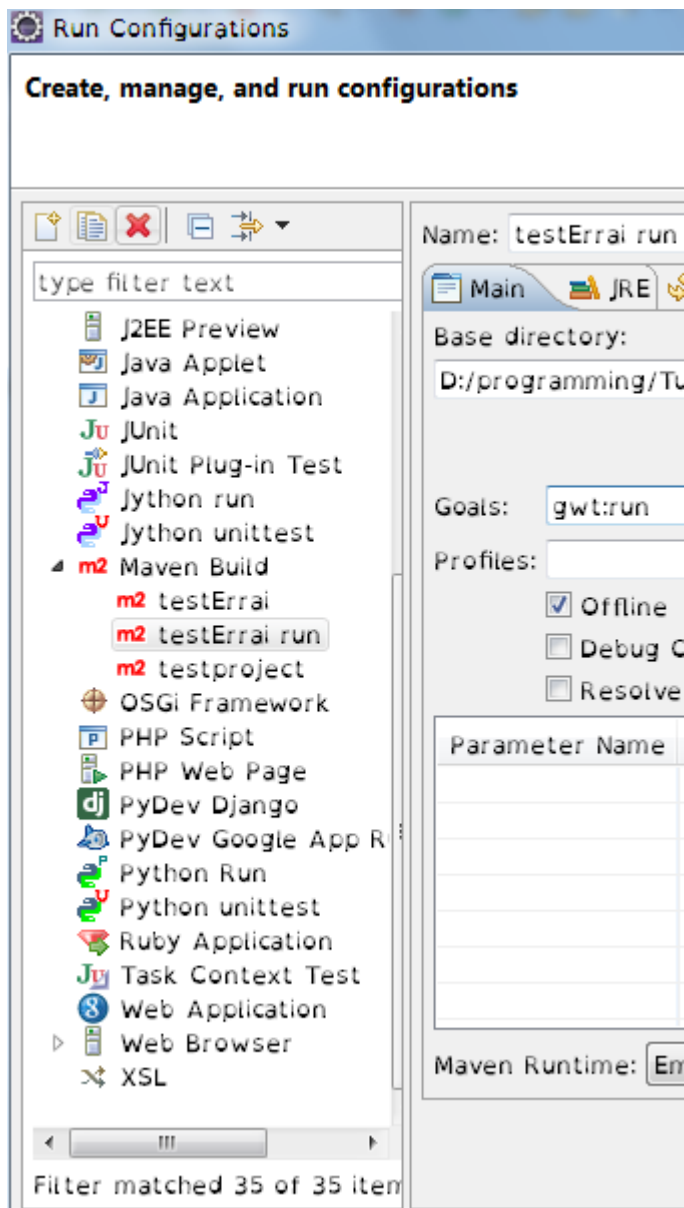
For building use install. And clean install for first cleaning up and then creating the files.

What to mention as well. If **Offline** is checked this means that no new or updates to dependencies will be downloaded. (Dependencies are other jars you need/want to use in the project example gwt 2.4.0 which is the gwt dependencies). So if you change your pom.xml file (the maven config file) you need to make sure offline isn't checked. Click apply (to save the config) and Run.



Now you have a maven run that you can reuse over and over again. You do this by: clicking the green ball arrow (this will show latest run projects, if what you looking for is not there select Run Configurations...) or select Run --> Run Configurations... And select your maven run config.





Basically you don't type `mvn` before what you want to do.

So there is how you setup your first Errai project.
[Next I will show how to update your maven project.](#)

Extra

Here is a way to deploy to your Jboss server instance.
Requires that you have a Jboss as 7 server running.

This extra will create a .war file that you can then deploy on your server.

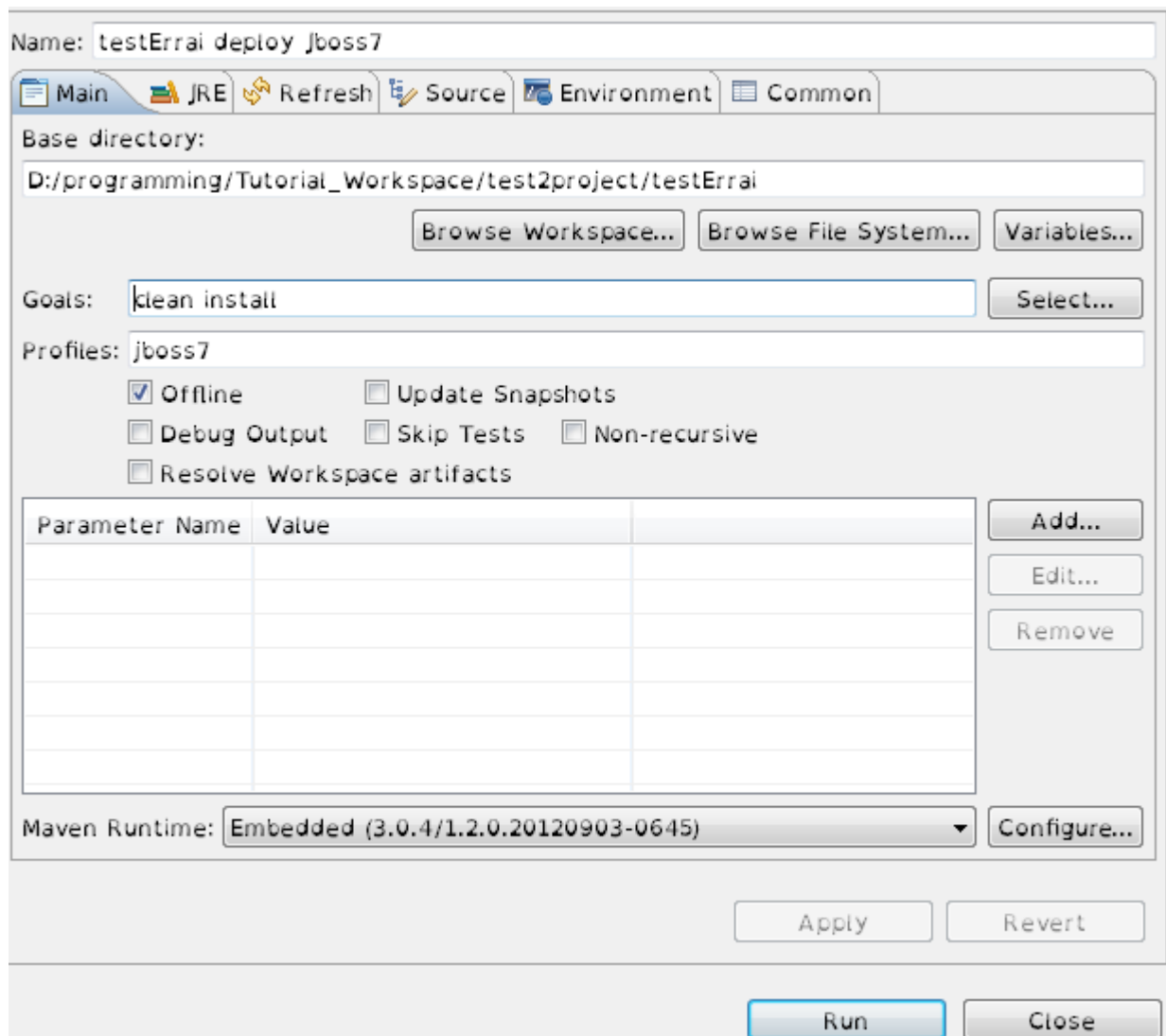
Extract Jboss package

What you need is a maven run configurations creating deployment:
Just like with the run and install you create a new run configuration.
Right click you project and select Run as --> 6 maven build...
Here you create a new run configuration.
Whats important is Goals: and Profiles:
Goals: clean install
Profiles: jboss7

Goal clean = cleans up the project and install creates the files.
Profile jboss7 = in your pom.xml there is a section with the name jboss7.
This means that you not only do the install stuff but also run the jboss7
stuff in the pom file. The result is a "projectname".war file.
The pom.xml part looks like this.

```
<profile>
  <id>jboss7</id>
...
</profile>
```

... = (symbolizing there is more code but im not writing it out, check you pom.xml file!).



Add to your Jboss server.

So after running your "deploy on jboss7" you should have .war file.
 If you look at the picture bellow you see on one of the installing lines the .war file is mentioned (mine is testErrai.war).

```
<terminated> testErrai deploy jboss7 [Maven Build] C:\Program Files\Java\jdk1.7.0_04\bin\javaw.exe (4 :
[INFO]
[INFO] --- maven-install-plugin:2.3.1:install (default-install) @ testErrai ---
[INFO] Installing D:\programming\Tutorial_Workspace\test2project\testErrai\target\testErrai.war
[INFO] Installing D:\programming\Tutorial_Workspace\test2project\testErrai\pom.xml to d:\data\m
[INFO]
[INFO] BUILD SUCCESS
[INFO]
[INFO] Total time: 1:41.335s
[INFO] Finished at: Thu Apr 04 18:46:04 CEST 2013
[INFO] Final Memory: 31M/178M
[INFO]
```

Jboss server adding

Now you have a war file that you can add to your server.
Make sure the server is started and your web management is setup for Jboss as 7.

Login on to your Jboss web management page,
Click **Manage Deployments**

JBoss Management - Mozilla Firefox

File Edit View History Bookmarks Tools Help

JBoss Management

localhost:9990/console/App.html#server-overview

JBoss Application Server 7.1

Server Status

Configuration

JVM

Subsystem Metrics

Datasources

JPA

Transactions

Web

Runtime Operations

OSGi

Deployments

Manage Deployments

Webservices

Standalone Server

Server: delux

Server configuration status. In some cases the confi

Server Configuration

The server configuration seems uptodate!

Code Name: Brontes

Server State: running

Extensions

Environment Properties

Name
org.jboss.as.clustering.infinispan
org.jboss.as.configadmin
org.jboss.as.connector
org.jboss.as.deployment-scanner
org.jboss.as.ee
org.jboss.as.ejb3
org.jboss.as.jaxrs
org.jboss.as.jdr

JBoss Application Server 7.1

▼ Server Status

Configuration
JVM

▼ Subsystem Metrics

Datasources
JPA
Transactions
Web

▼ Runtime Operations

OSGi

▼ Deployments

Manage Deployments

Webservices

Deployments

Deployments

Name	Runtime Name	Enabled	En/Dis
No items!			

Now click Add Content.

Upload

Step1/2: Deployment Selection

Please choose a file that you want to deploy.

Browse...

Next >>

Cancel

Browse... And find your .war file (remember from before is in the project target folder) and choose to open it.

Upload

Step1/2: Deployment Selection

Please choose a file that you want to deploy.

Browse...

Next >>

Cancel

Click Next.

Upload

Step 2/2: Verify Deployment Names

Key: zozR9gjh0TXr1EL7ZVfi0le5sH

Name: testErrai.war

Runtime Name: testErrai.war

Save

Cancel

Click Save.

JBoss Application Server 7.1

▼ Server Status

Configuration

JVM

▼ Subsystem Metrics

Datasources

JPA

Transactions

Web

▼ Runtime Operations

OSGi

▼ Deployments

Manage Deployments

Webservices

Deployments

Deployments

Name	Runtime Name	Enabled	En/D
testErrai.war	testErrai.war		<div>Ena</div>

Now you should have something like this added to your Jboss web management page. What you need to do now is click Enable. This will enable your webapp.

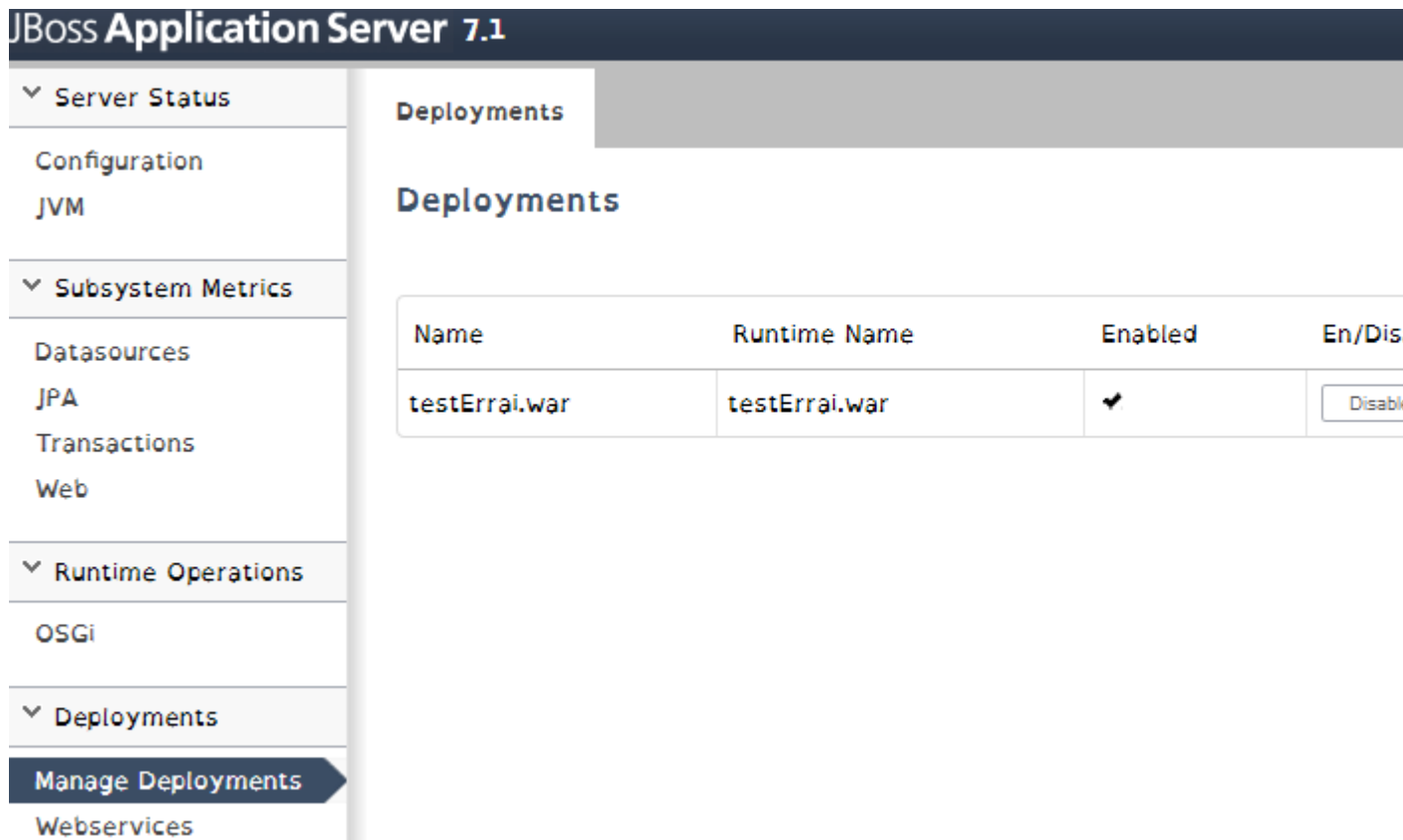


Enable testErrai.war?

Confirm Cancel

Click Confirm.





The screenshot shows the JBoss Application Server 7.1 web console. On the left is a navigation menu with sections: Server Status (containing Configuration and JVM), Subsystem Metrics (containing Datasources, JPA, Transactions, and Web), Runtime Operations (containing OSGi), Deployments (containing Manage Deployments and Webservices), and a highlighted Manage Deployments button. The main content area is titled 'Deployments' and contains a table with the following data:

Name	Runtime Name	Enabled	En/Dis
testErrai.war	testErrai.war	<input checked="" type="checkbox"/>	Disable

Now you should have something like this. Notice that enabled is checked and en/Disable is only showing Disable.
Now your web project is up and running. Congratulations!

If you don't know how to get to it. Try: localhost:5657/"projectname"
:5657 = this is my port for webapps. You need to know which port you are running your Jboss web on.

Im using the Standalone. That means in my Jboss as 7 folder in using
standalone\configuration\standalone.xml
And in here my port settings are set.
`<socket-binding name="http" port="5657"/>`

If you have been using same name as me ErraiTest
the url is: <http://localhost:5657/testErrai>
Notice that The first E is big. If you like me wrote it big it has to be big.
This is because in the pom.xml the name and artifactid is spelt like this:

```
<name>testErrai</name>
<groupId>edu.erraitesting</groupId>
<artifactId>testErrai</artifactId>
```

If you now change the names to:
`<name>testerrai</name>`

```
<groupId>edu.erraitesting</groupId>  
<artifactId>testerrai</artifactId>
```

And redeploy on the server (redo everything of deploy to jboss7).
The url will be <http://localhost:5657/testerrai>