Conceptos previos necesarios:

Antes de empezar recapitulamos algunos conceptos necesarios para aquellas personas que no hayan trabajado con un entorno similar.

Como se comenta en el documento descriptivo de la arquitectura en :

https://github.com/thefinerthingsclub/finerleague/blob/master/docs/Informaci%C3%B3n%20t%C3%A9cnica.pdf

Vamos a trabajar con una Arquitectura MVC orientada a Servicios.

http://www.evaluandosoftware.com/soa-arquitectura-orientada-a-servicios/

Para aquellos que no hayan tratado con servicios web recomiendo este enlace donde se explica para qué sirven y la diferencia entre los SOAP y los REST.

https://www.adictosaltrabajo.com/tutoriales/soavs-soap-rest/

En nuestro proyecto vamos a trabajar con REST y ayudándonos de Spring que facilita enormemente su creación y uso.

En cuanto a la BBDD utilizada vamos a usar una BBDD No SQL llamada MongoDB.

MongoDB es una base de datos orientada a documentos. Esto quiere decir que en lugar de guardar los datos en registros, guarda los datos en documentos. Estos documentos son almacenados en BSON, que es una representación binaria de JSON.

Para más información:

https://www.genbetadev.com/bases-de-datos/mongodb-que-es-como-funciona-y-cuando-podemos-usarlo-o-no

Además como comenta el documento de arquitectura utilizaremos el framework de Spring, Spring Data MongoDb que nos facilitara el acceso a estos “documentos”. Es similar a utilizar cualquier implementación de un JPA como Hibernate, MyBatis, Spring Data JPA, etc…

Otra cosa que se da por supuesta es que conocemos SPRING. Para el que no lo conozca recomendamos el siguiente enlace:

Conceptos básicos y ejemplo de IOC:

http://migranitodejava.blogspot.com.es/2011/06/introduccion-spring.html

¿Por qué utilizar Spring?

https://rekkeb.wordpress.com/2012/05/13/por-que-spring-simplifica-el-desarrollo-de-nuestras-aplicaciones-java/

Programación con Aspectos de Spring:

http://www.jtech.ua.es/j2ee/publico/spring-2012-13/apendice_AOP-apuntes.html

Por último nos falta comentar para quien no haya trabajado repositorios de datos donde

compartir el código desarrollado por el equipo, vamos a trabajar con GIT.

Difiere de los repositorios de datos tradicionales como SVN o CVS en varios aspectos. Para simplificarlo todos tendremos un repositorio local donde hacer commit de nuestros desarrollos, pero para poder compartirlos será necesario subirlos(push) a una rama remota que comparte el proyecto. Para más información:

https://git-scm.com/book/es/v1/Empezando-Fundamentos-de-Git

y recomendamos para la forma de trabajo la propuesta por Gitflow:

http://aprendegit.com/que-es-git-flow/

Con todo esto y un IDE de desarrollo ya estamos listos para empezar a trabajar.

Instalación del entorno:

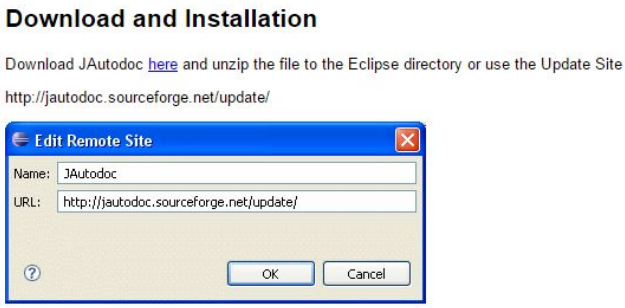
1.-Tenemos que descargar un IDE, como ejemplo coy a configurar un eclipse. Para el intelliJ IDEA no deja de ser similar.

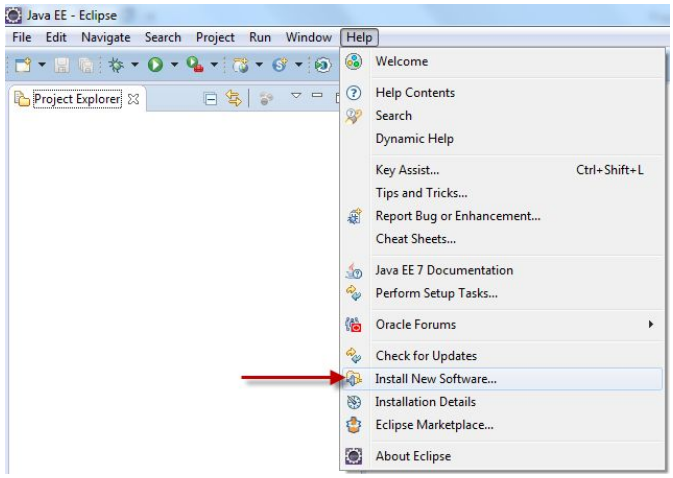
https://eclipse.org/downloads/

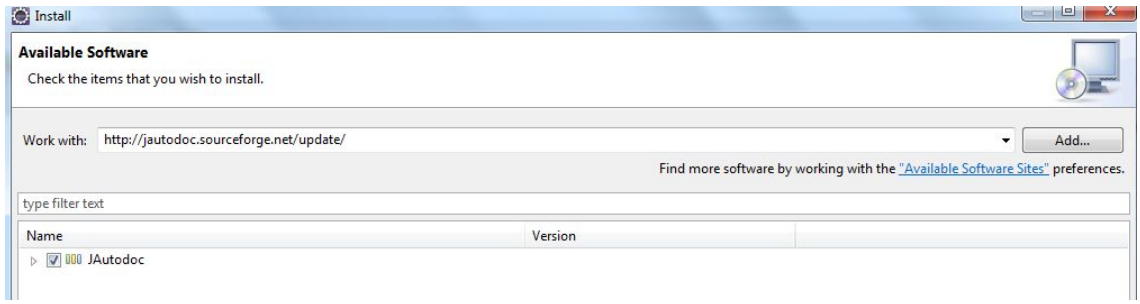
En mi caso tengo instalada la versión MARS, necesaria como mínimo para trabajar con los plugins de GIT necesarios.

Es necesario el formateo de código para que no haya problemas de integración con GIT además de para seguir con las buenas practicas recomendables en programación.

Para instalar el plugin en eclipse:

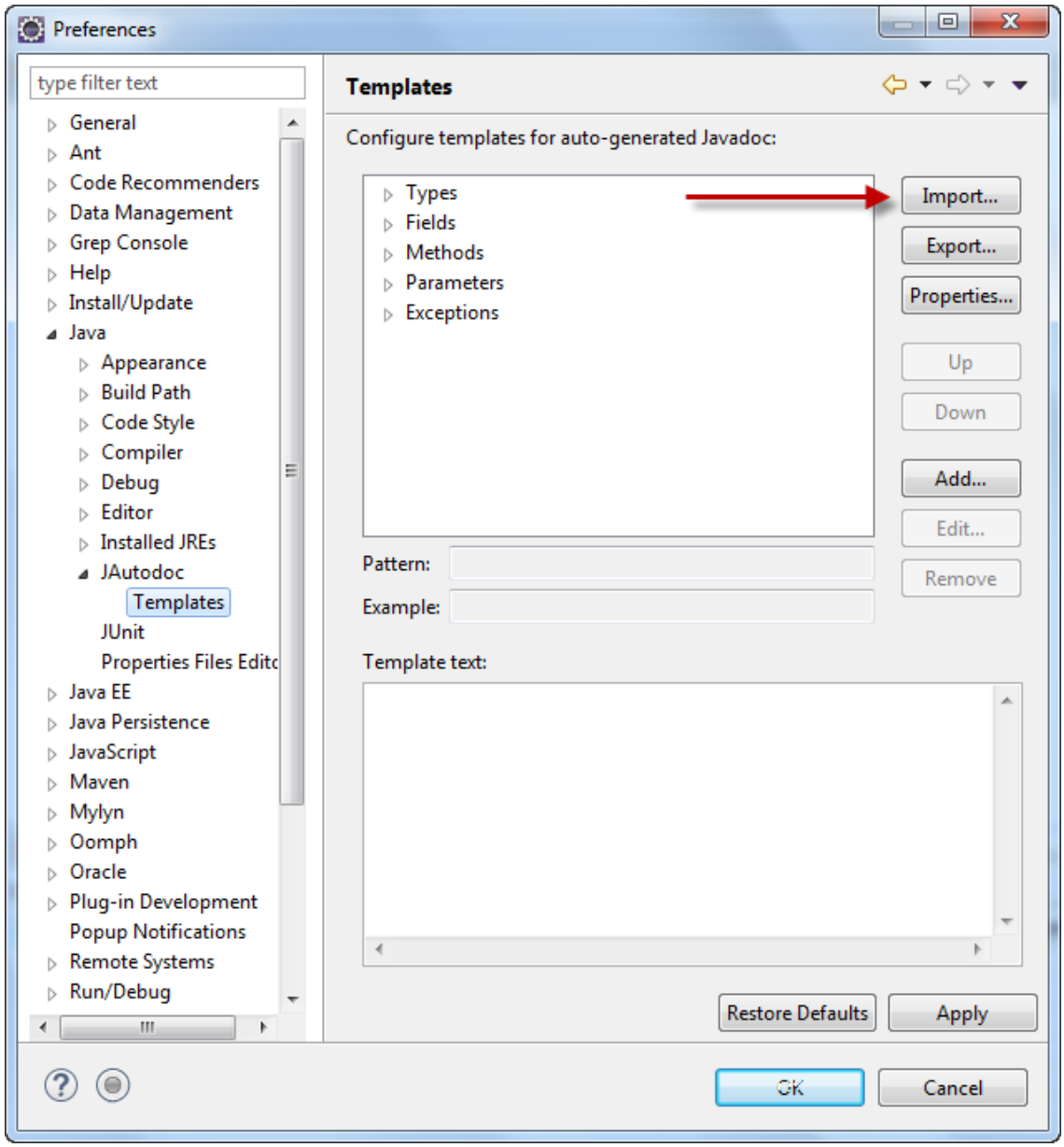






Finalizamos la instalación e importamos la template del proyecto:

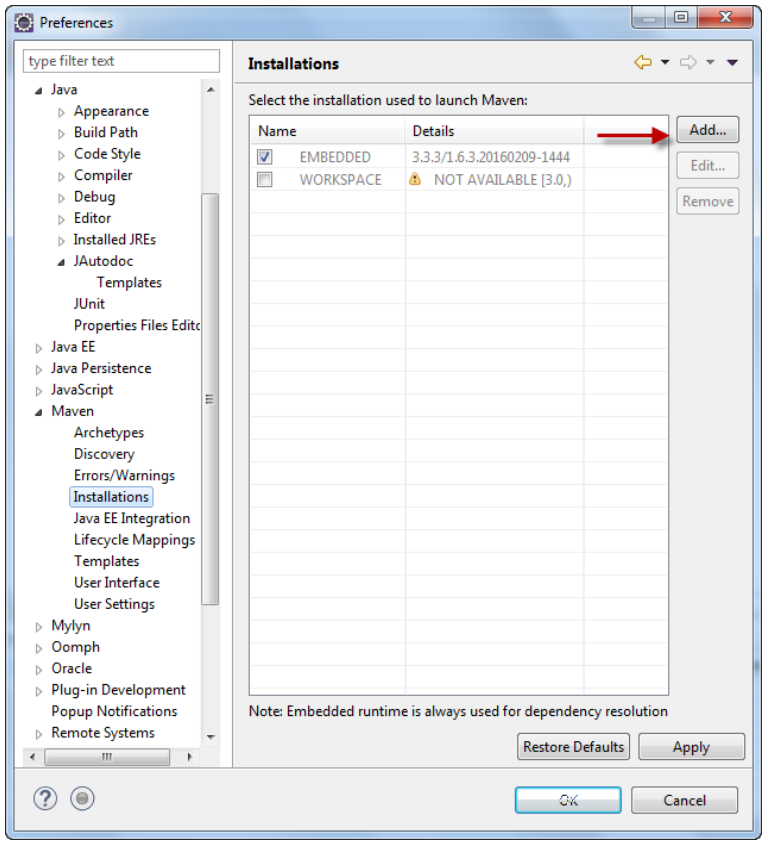
https://github.com/thefinerthingsclub/finerleague/blob/master/docs/CodeStyle.xml



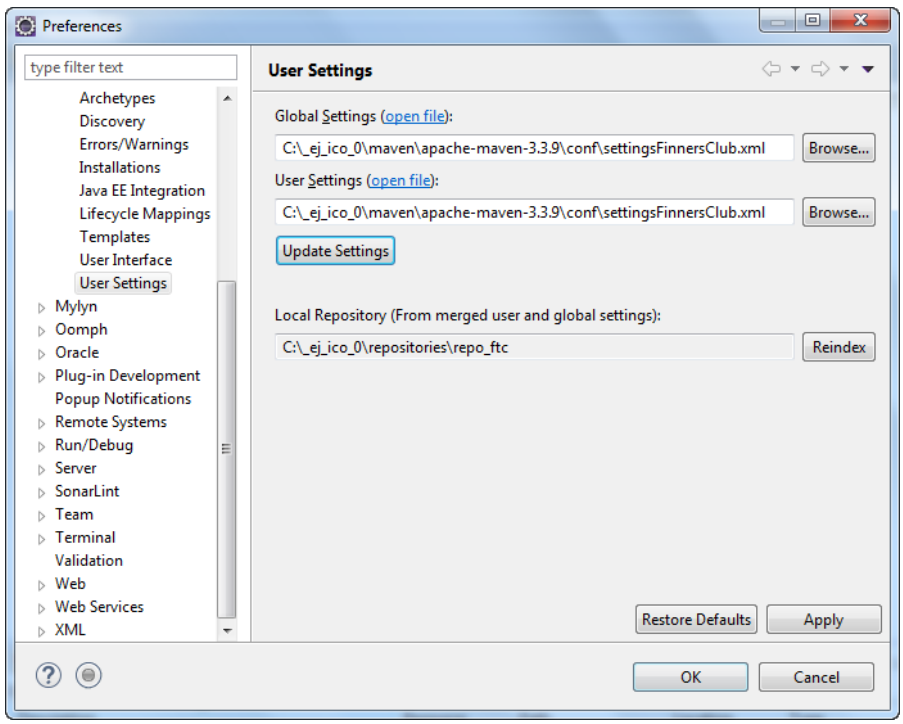
Por otro lado nos descargamos maven :

https://maven.apache.org/download.cgi

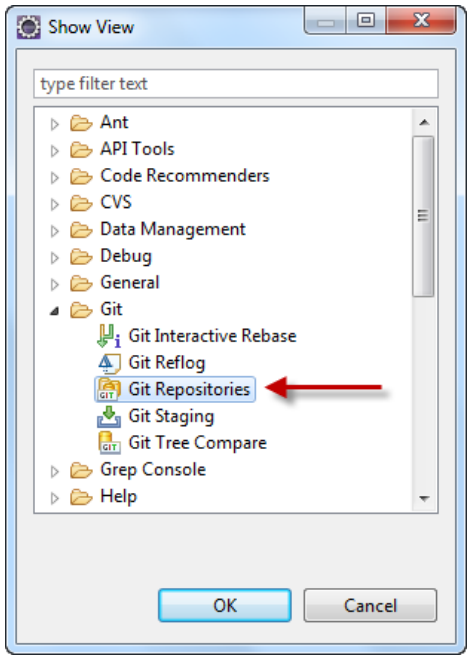
y configuramos el settings desde eclipse:

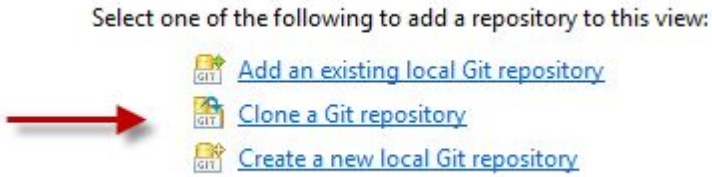


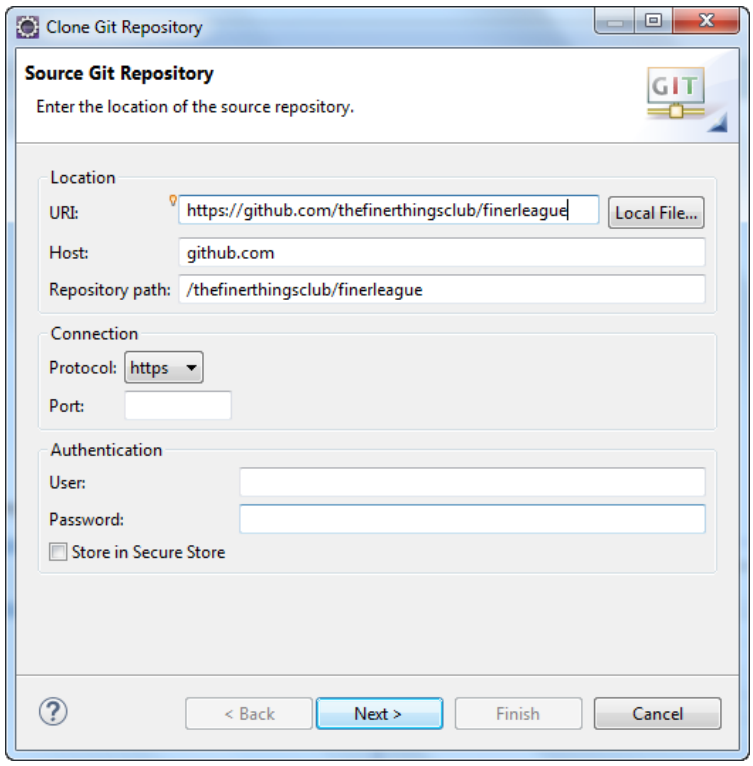
Y añadimos el settings(yo he cambiado la ruta a otro directorio en el archivo settings.xml):



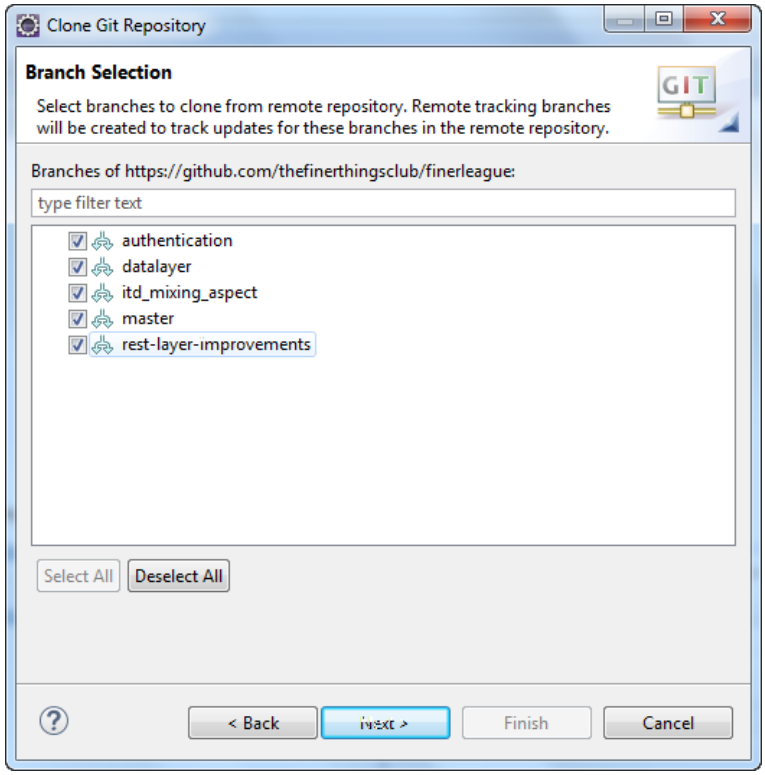
Con esto ya vamos a conectarnos a git e importar el proyecto:


Show View
type filter text
Ant
API Tools
Code Recommenders
CVS
Data Management
Debug
General
Git
Git Interactive Rebase
Git Reflog
Git Repositories
Git Staging
Git Tree Compare
Grep Console
Help
OK
Cancel

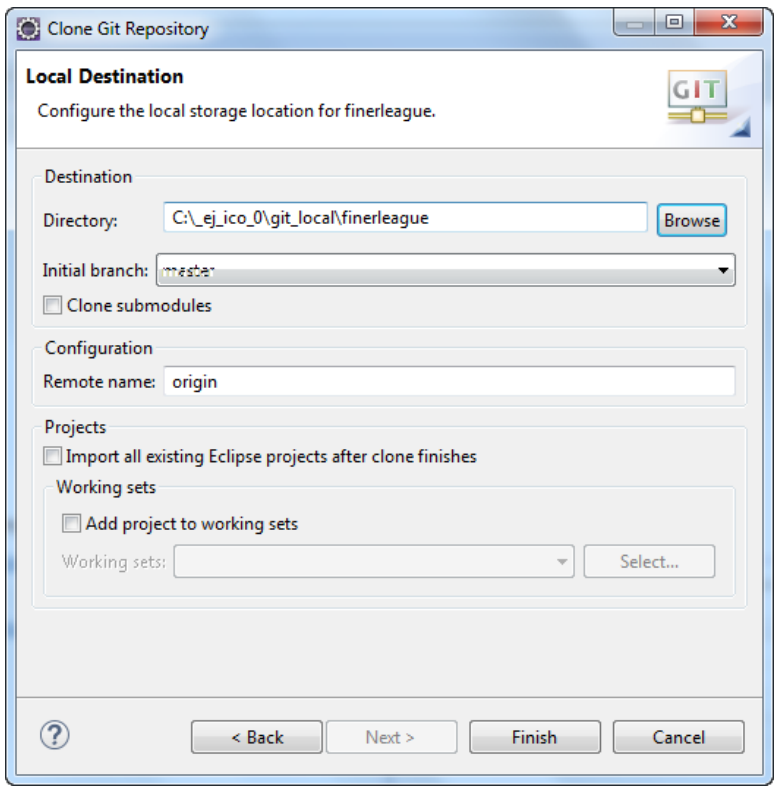

Select one of the following to add a repository to this view:
Add an existing local Git repository
Clone a Git repository
Create a new local Git repository


Clone Git Repository
Source Git Repository
Enter the location of the source repository.
GIT
Location
URI:
https://github.com/thefinerthingsclub/finerleague
Local File...
Host:
github.com
Repository path:
/thefinerthingsclub/finerleague
Connection
Protocol:
https
Port:
Authentication
User:
Password:
Store in Secure Store
< Back
Next >
Finish
Cancel

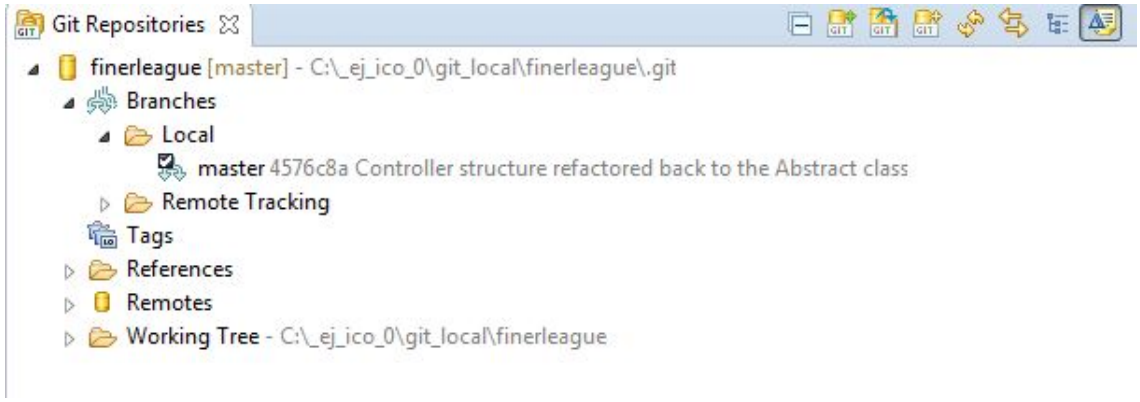
Y ahora seleccionamos que ramas bajarnos:



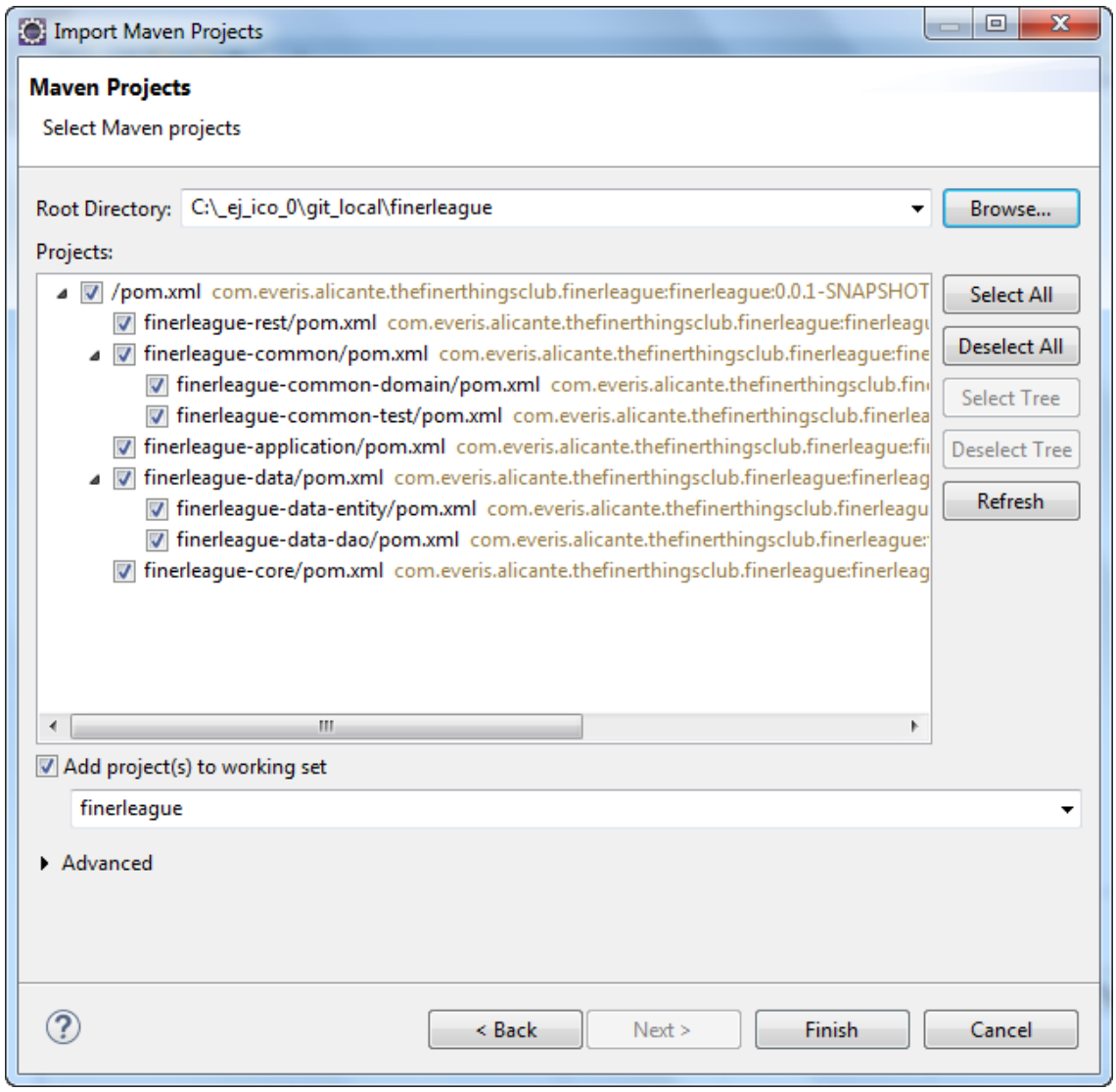
Y yo he cambiado el directorio del git local en el paso siguiente:



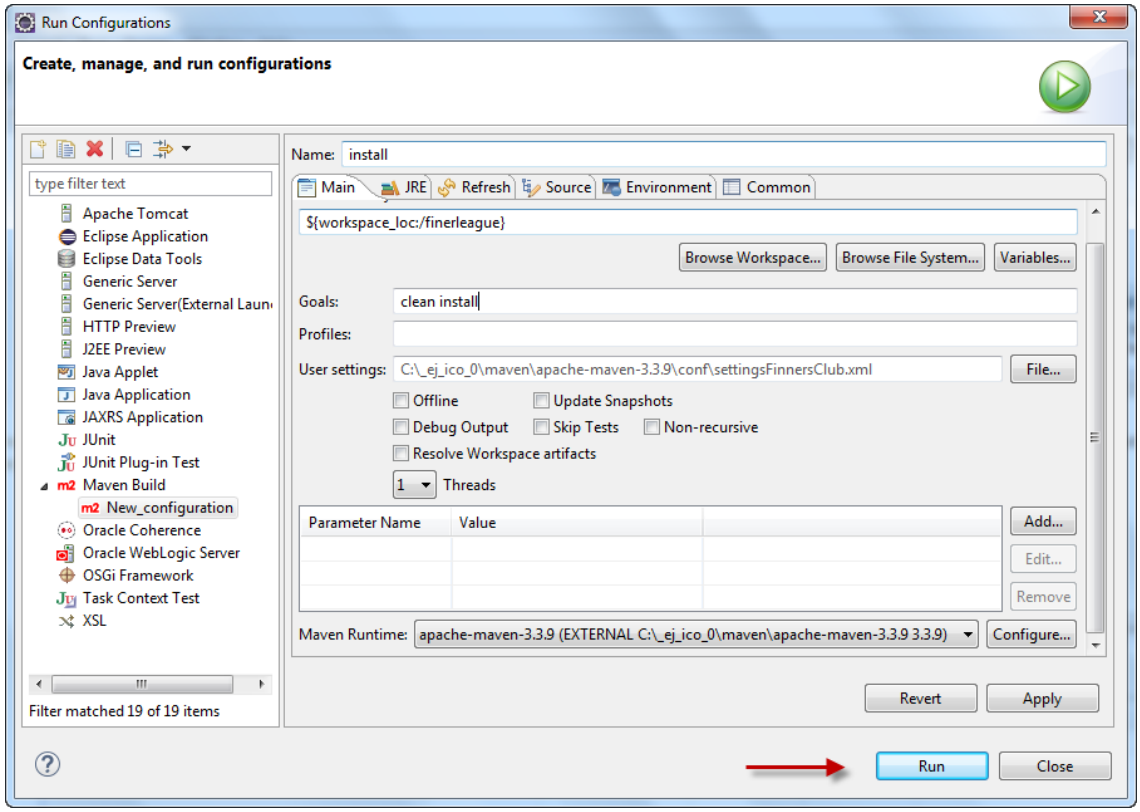
Y ya tenemos el proyecto sincronizado y descargado:



Y lo importamos como proyecto maven:



Lanzamos maven y listo:




```
[INFO] finerleague-data ..................................... SUCCESS [  0.029 s]
[INFO] finerleague-data-entity .............................. SUCCESS [  1.103 s]
[INFO] finerleague-common-test .............................. SUCCESS [  1.843 s]
[INFO] finerleague-data-dao ................................. SUCCESS [  1.313 s]
[INFO] finerleague-core ..................................... SUCCESS [  0.519 s]
[INFO] finerleague-rest ..................................... SUCCESS [  1.462 s]
[INFO] finerleague-application .............................. SUCCESS [  6.069 s]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 34.323 s
[INFO] Finished at: 2017-01-22T20:57:54+01:00
[INFO] Final Memory: 47M/256M
[INFO] ------------------------------------------------------------------------
```