EJEMPLOS EN HADOOP

EJEMPLO 1:

Veamos un ejemplo sencillo de Hadoop. Hadoop proporciona el siguiente ejemplo MapReduce archivo jar, que proporciona la funcionalidad básica de MapReduce y puede utilizarse para calcular, valores como: PI, recuento de palabras en una determinada lista de archivos, etc.

Vamos a tener un directorio de entrada donde vamos a ingresar varios archivos y nuestro ejemplo consiste en contar el número total de palabras en los archivos.
Para calcular el número total de palabras, no es necesario escribir nuestro MapReduce, siempre el archivo .jar contiene la implementación de recuento de palabras.

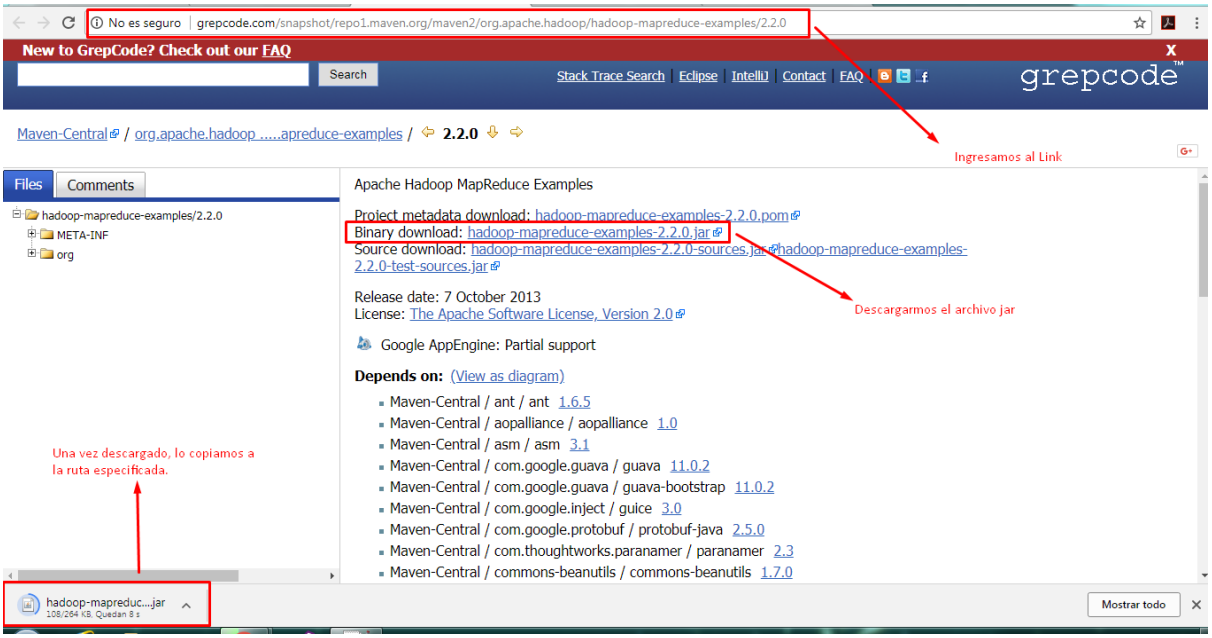
El archivo .jar lo podemos ejecutar con la siguiente línea en la terminal:

```
$ hadoop jar $HADOOP_HOME/share/hadoop/mapreduce/hadoop-mapreduceexamples-2.2.0.jar
```

Hay que tener en cuenta que el archivo .jar debe existes en la ruta ya especificado, de lo contrario no se podrá realizar el ejemplo.

En caso de no contar con el archivo `hadoop-mapreduceexamples-2.2.0.jar` lo podemos descargar desde el siguiente sitio:

http://grepcode.com/snapshot/repo1.maven.org/maven2/org.apache.hadoop/hadoop-mapreduce-examples/2.2.0



Paso 1

Crear contenido temporal archivos en el directorio de entrada. Se puede crear este directorio de entrada en cualquier lugar:

```
$                              mkdir                                  input
$              cp               $HADOOP_HOME/*.txt                   input
$ ls -l input
```

Le dará los siguientes archivos en el directorio de entrada:

```
total 24
-rw-r--r-- 1 root root 15164 Feb 21 10:14 LICENSE.txt
-rw-r--r-- 1 root root   101 Feb 21 10:14 NOTICE.txt
-rw-r--r-- 1 root root  1366 Feb 21 10:14 README.txt
```

Estos archivos se han copiado del Hadoop instalación directorio de inicio. Para el experimento, que puede tener diferentes y grandes conjuntos de archivos.

Paso 2

Vamos a comenzar el proceso Hadoop para contar el número total de palabras en todos los archivos disponibles en el directorio de entrada, de la siguiente manera:

```
$ hadoop jar $HADOOP_HOME/share/hadoop/mapreduce/hadoop-
mapreduceexamples-2.2.0.jar  wordcount input ouput
```

Paso 3

El paso 2 hará el procesamiento necesario y guardar el resultado del producción/parte-r00000 archivo, que se puede comprobar mediante:

```
$cat output/*
```

Para ver una lista de todas las palabras junto con su número total disponible en todos los archivos disponibles en el directorio de entrada.

```
"AS       4
"Contribution" 1
"Contributor" 1
"Derivative 1
"Legal 1
"License"       1
"License");      1
"Licensor"       1
"NOTICE”         1
"Not      1
"Object"         1
"Source”         1
"Work”    1
"You"     1
"Your")   1
"[]"      1
"control"        1
"printed         1
"submitted"      1
(50%)     1
(BIS),    1
(C)       1
(Don't)   1
```

```
(ECCN)    1
(INCLUDING      2
(INCLUDING,     2
.............
```

NOTA: Podemos intentar otros ejemplos en los que se usa el mismo archivo .jar, simplemente ejecute los siguientes comandos para comprobar apoya los programas funcionales de MapReduce hadoop mapreduce de ejemplos-2.2.0 .jar.

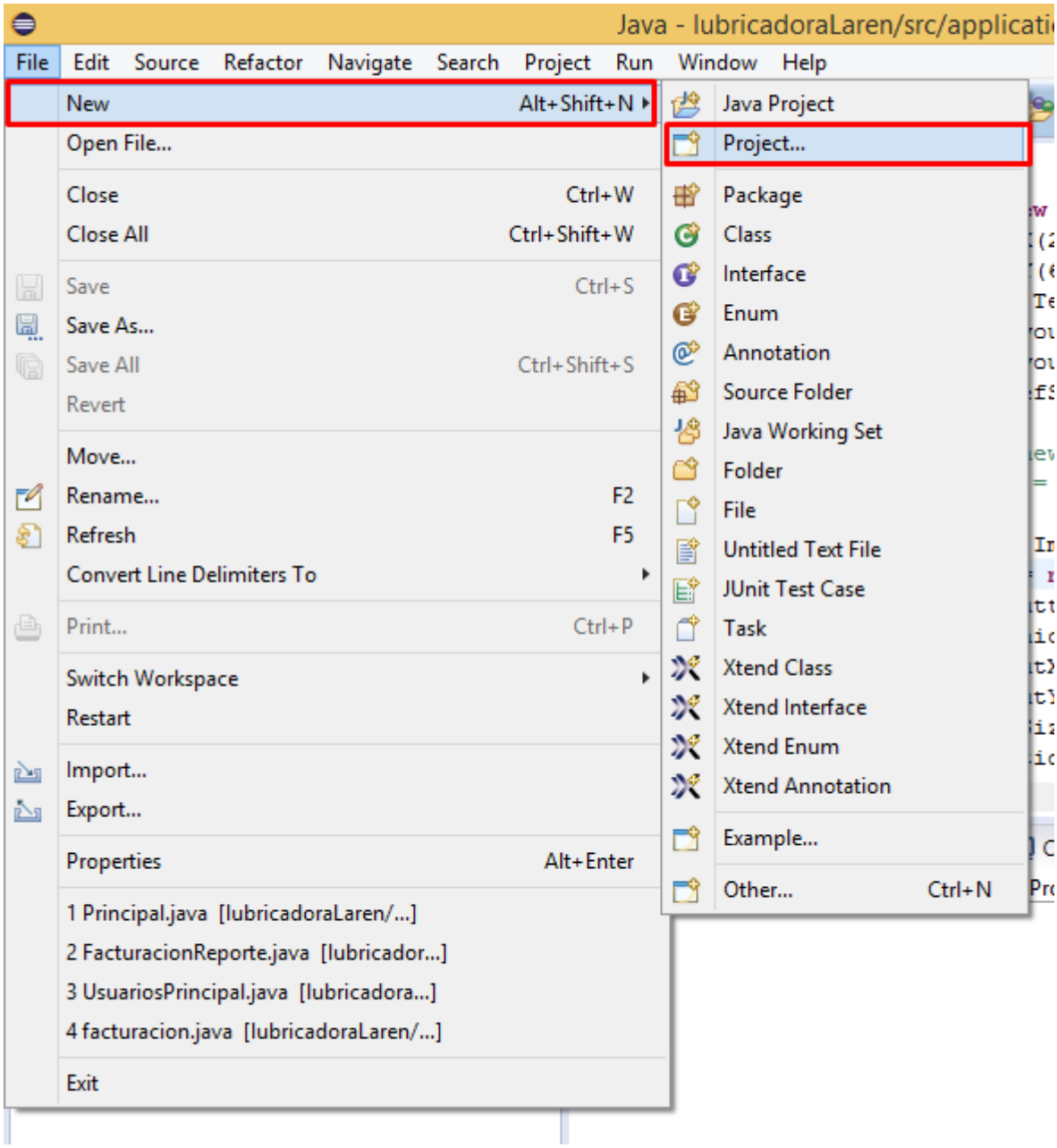
https://www.tutorialspoint.com/es/hadoop/hadoop_enviornment_setup.htm

EJEMPLO 2

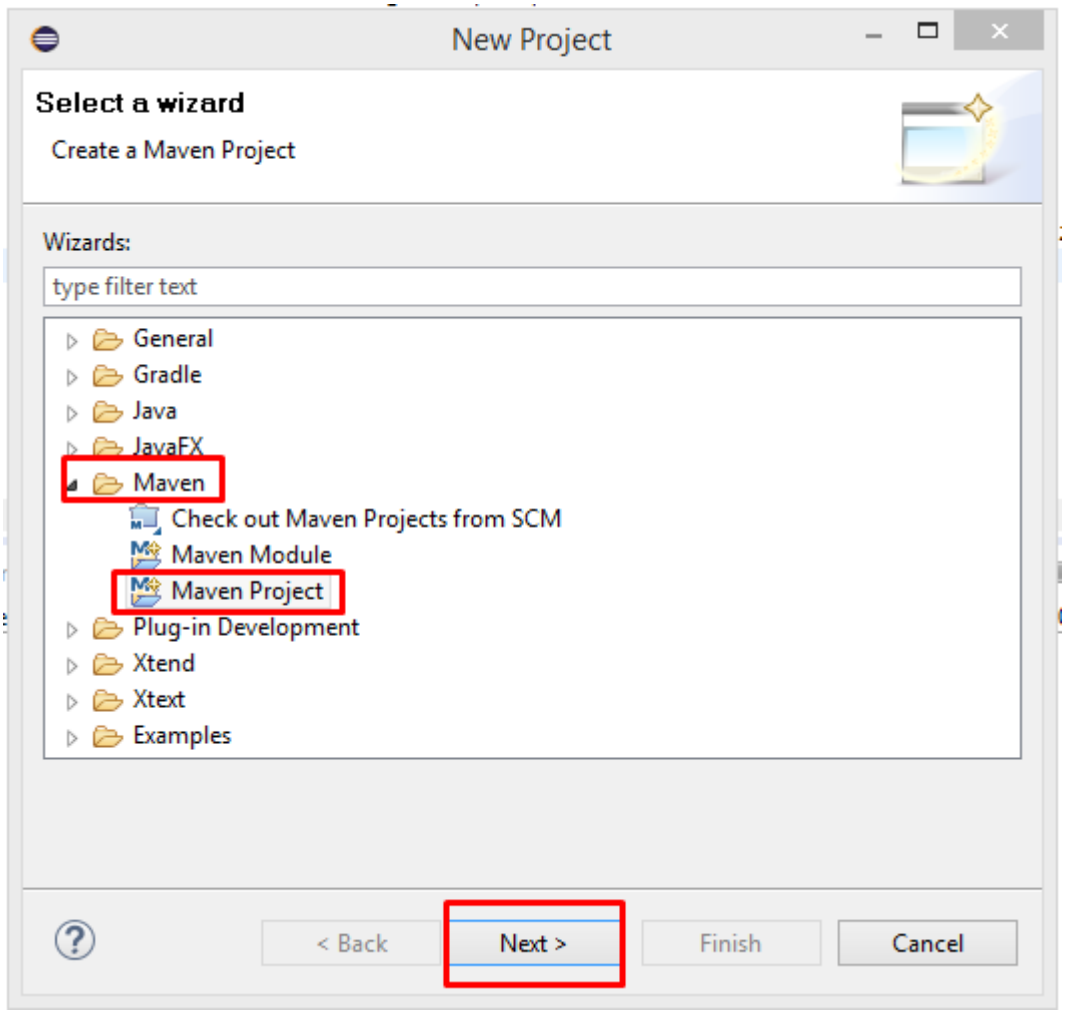
Para iniciar con este ejemplo debemos descargar los siguientes archivos:
datos de pruebas.
Debemos crear un proyecto en Maven con las siguientes clases:
Terremotos
TerremotosMapper
TerremotosReducer

Paso 1

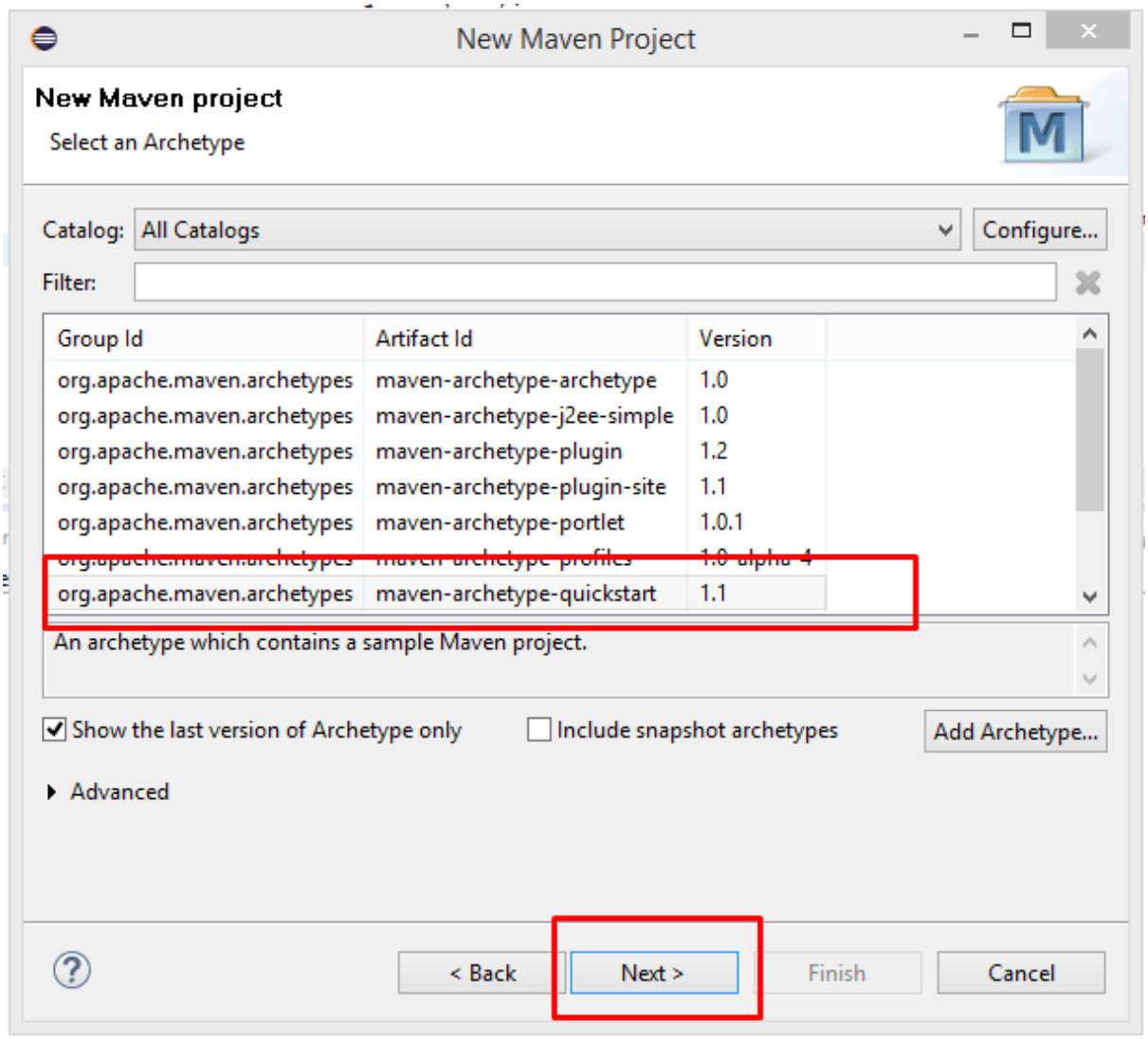
Abrimos el eclipse y creamos un nuevo proyecto:



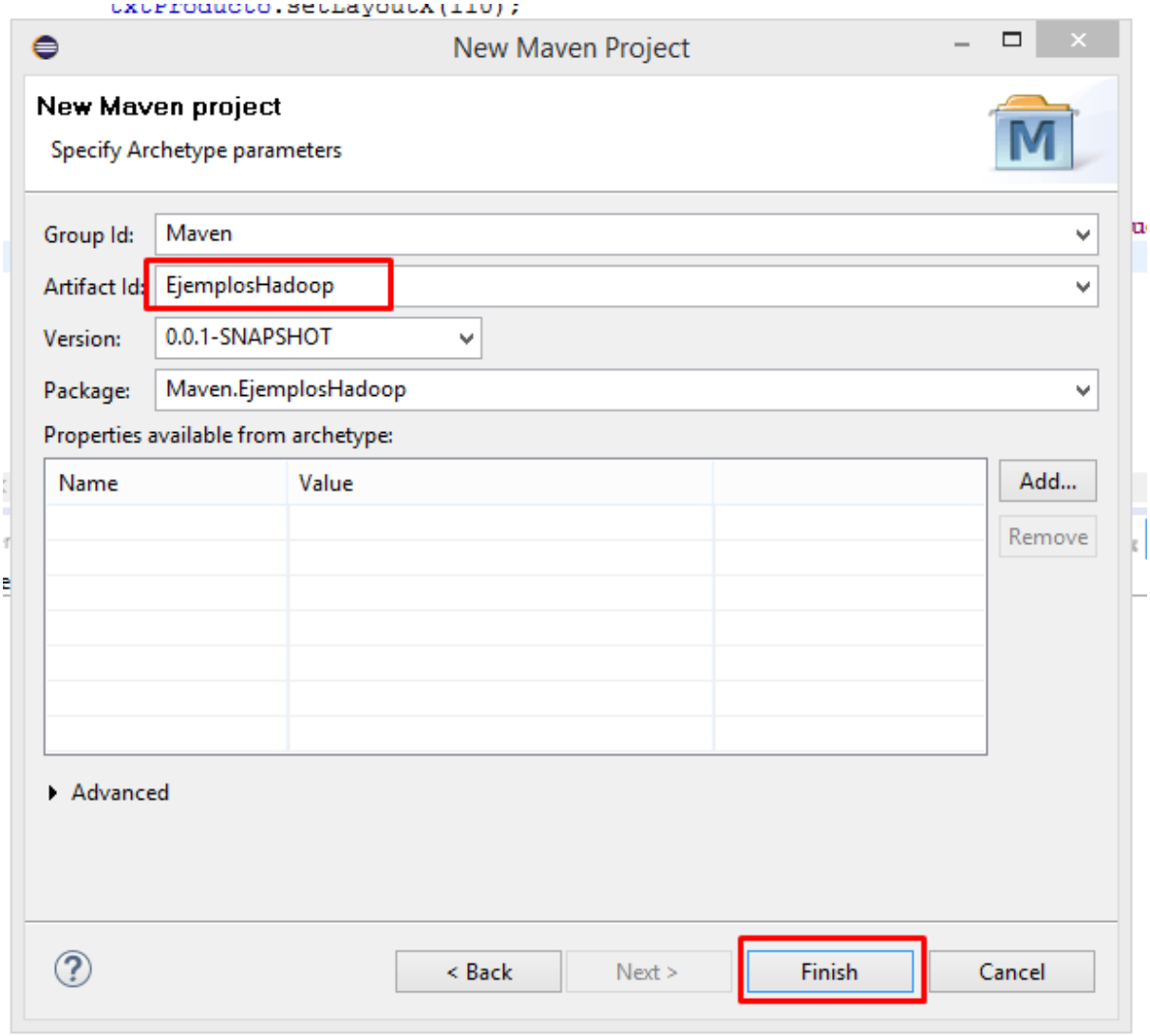
Elegimos la opción de Maven:



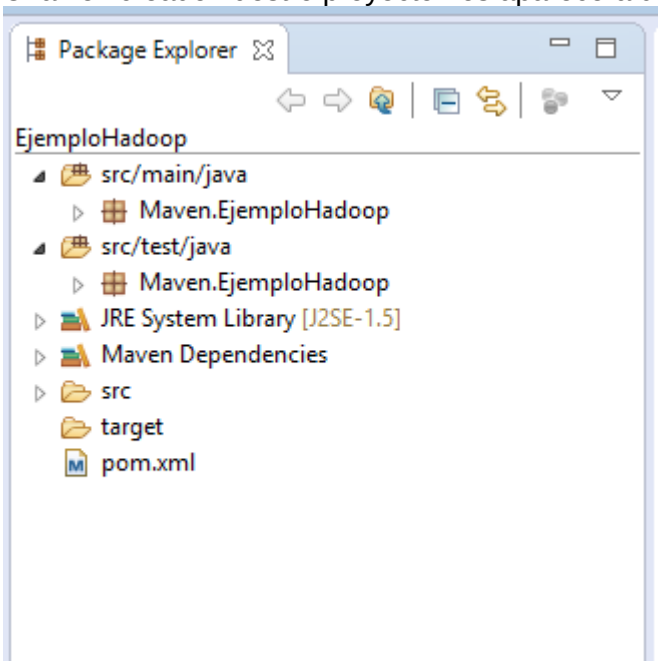
Clic en Next nuevamente, aparecerá la siguiente pantalla en la cual escogeremos la siguiente opción:



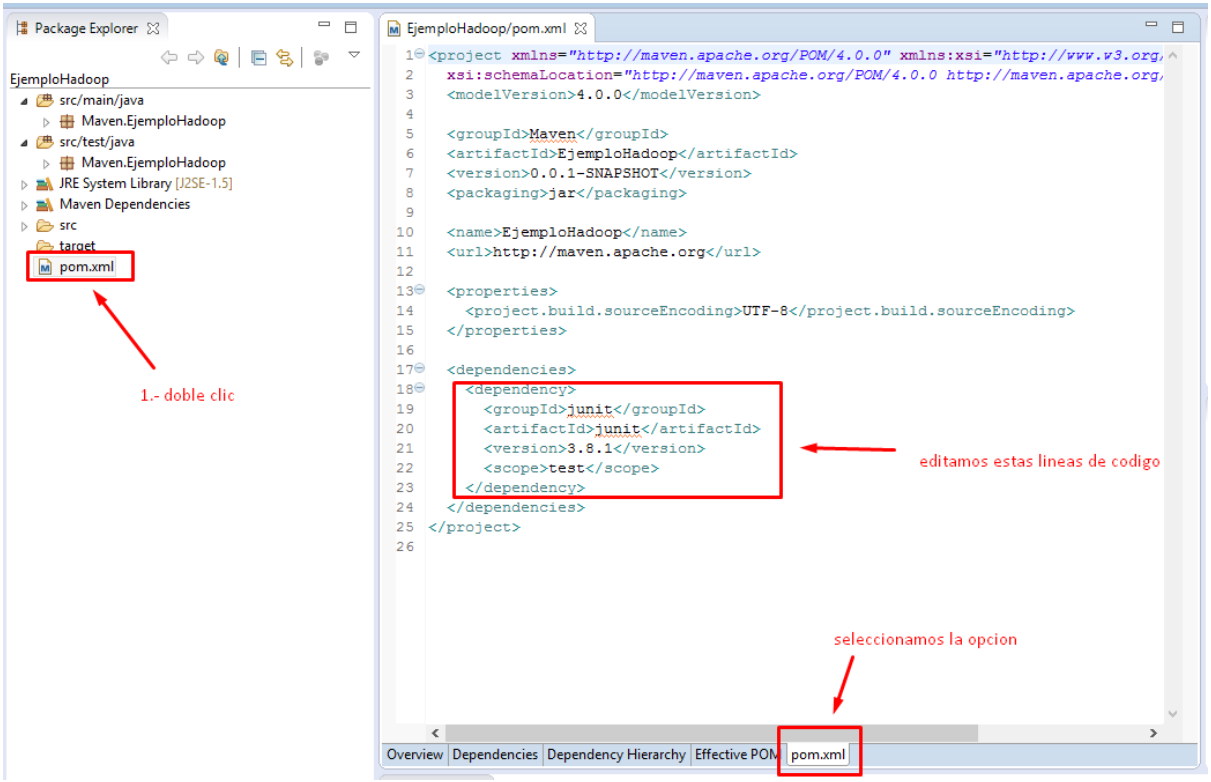
Luego le damos un nombre a los proyectos y finalizamos:



Una vez creado nuestro proyecto nos aparecerá de la siguiente manera:



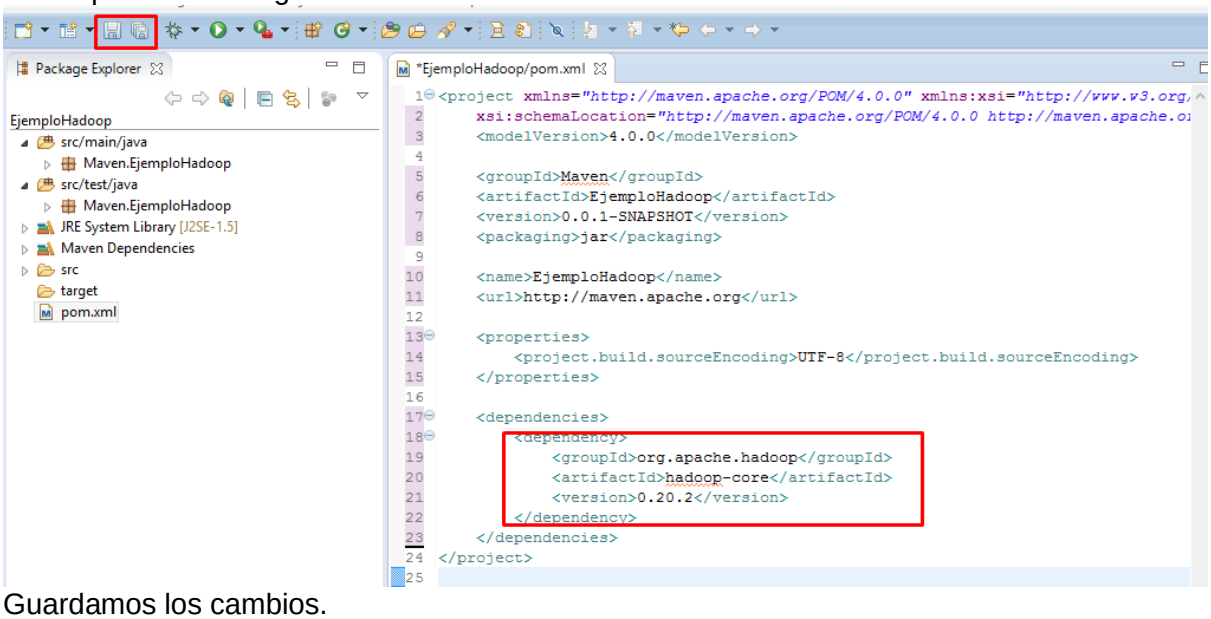
Abrimos el archivo pom.xml



Procedemos a editar el código colocando las siguientes líneas:

```
<dependency>
    <groupId>org.apache.hadoop</groupId>
    <artifactId>hadoop-core</artifactId>
    <version>0.20.2</version>
</dependency>
```

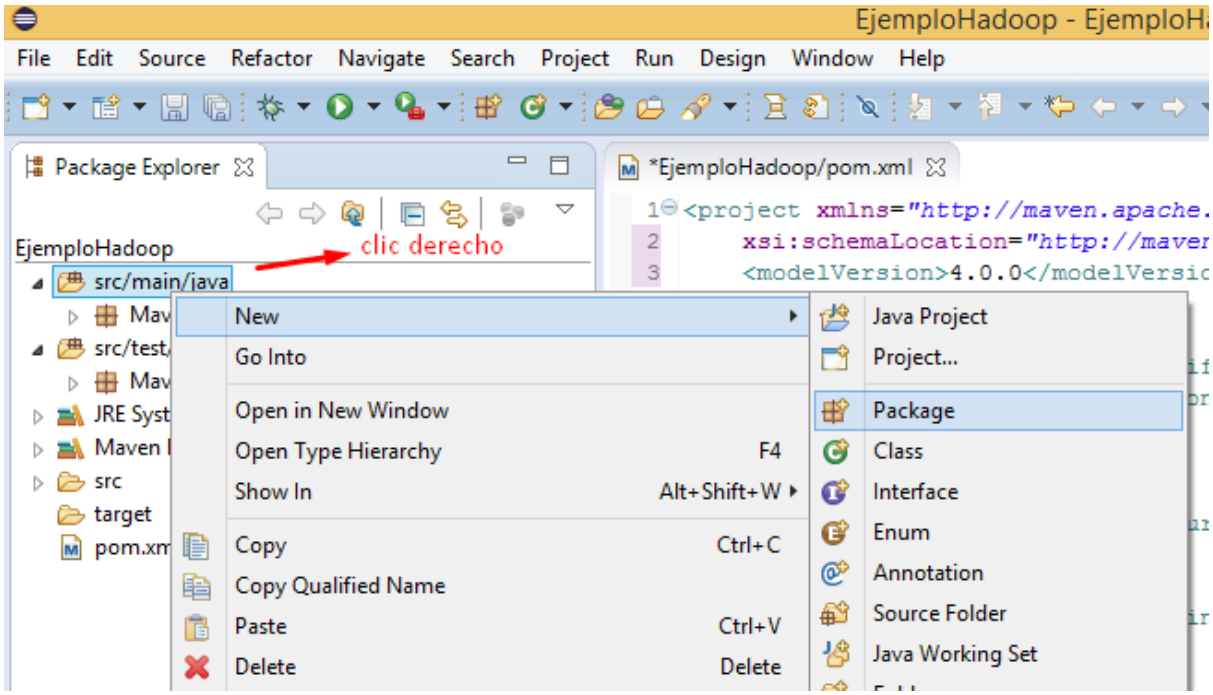
Debe quedar de la siguiente manera:



Guardamos los cambios.

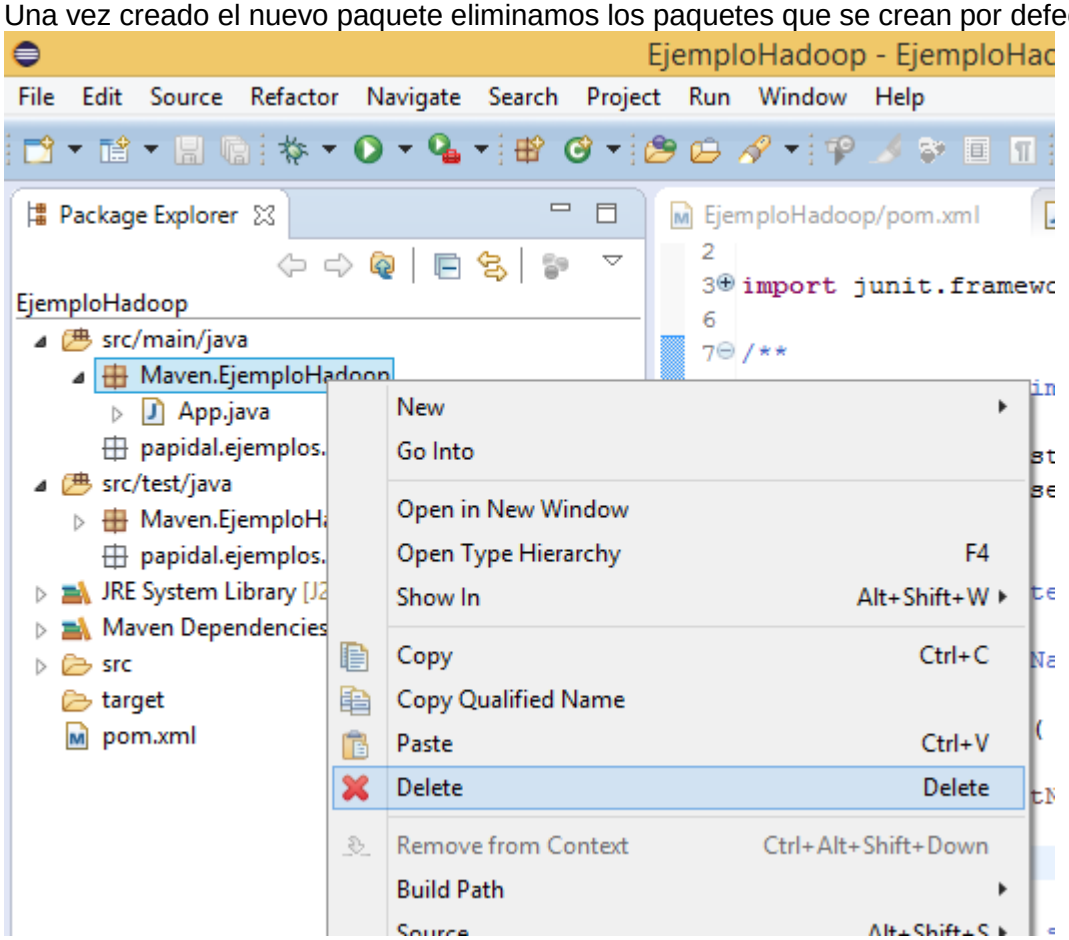
Alan Reyes Bacusoy

Una vez realizado este cambio procedemos a crear el siguiente paquete:



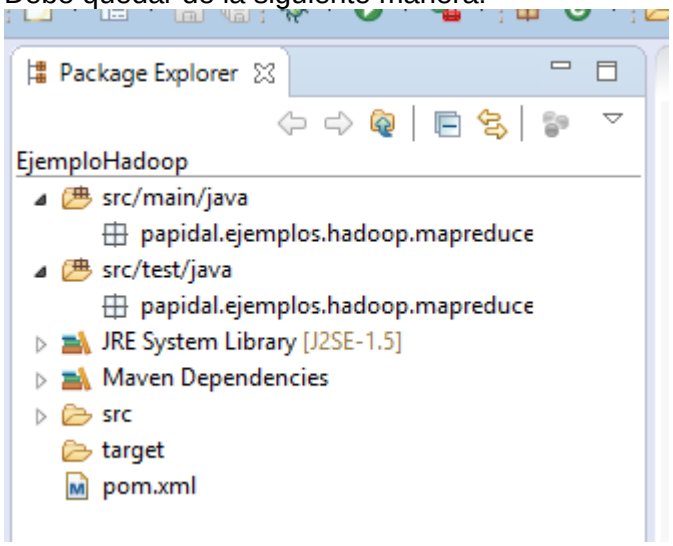
papidal.ejemplos.hadoop.mapreduce

Una vez creado el nuevo paquete eliminamos los paquetes que se crean por defecto:

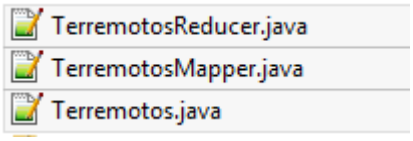


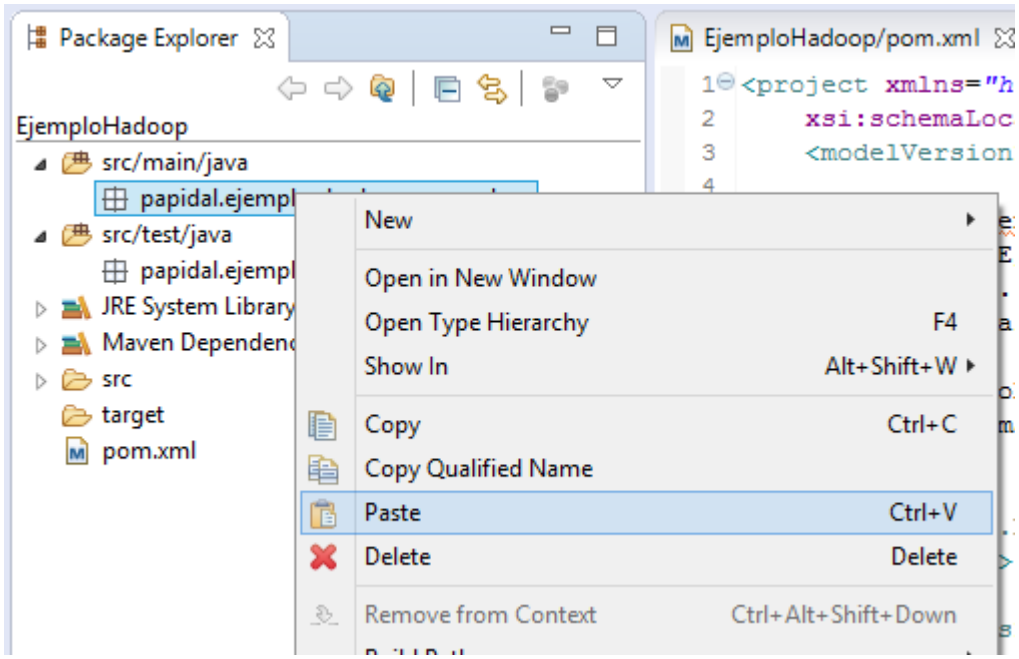
Alan Reyes Bacusoy

Debe quedar de la siguiente manera:

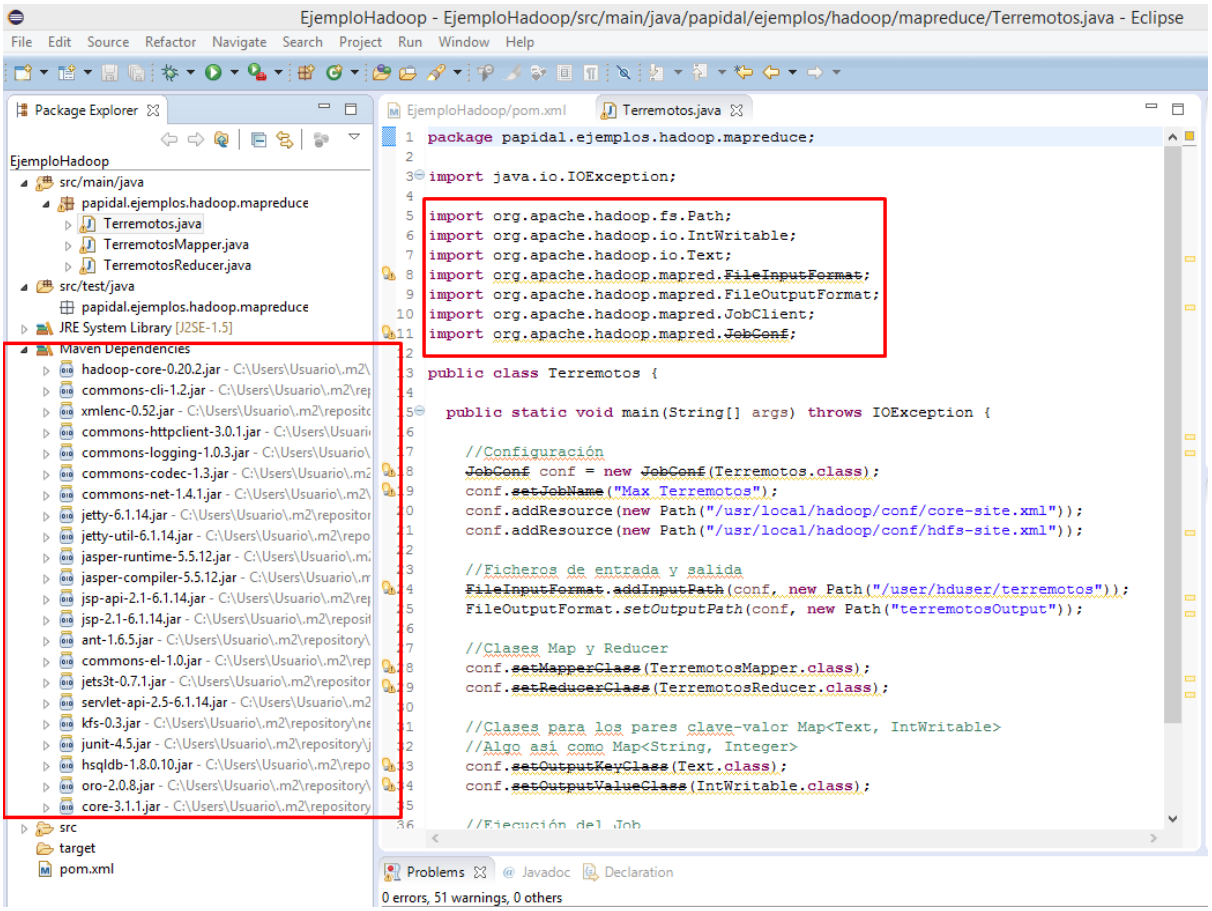


Copiamos las clases descargadas, dentro de nuestro paquete:



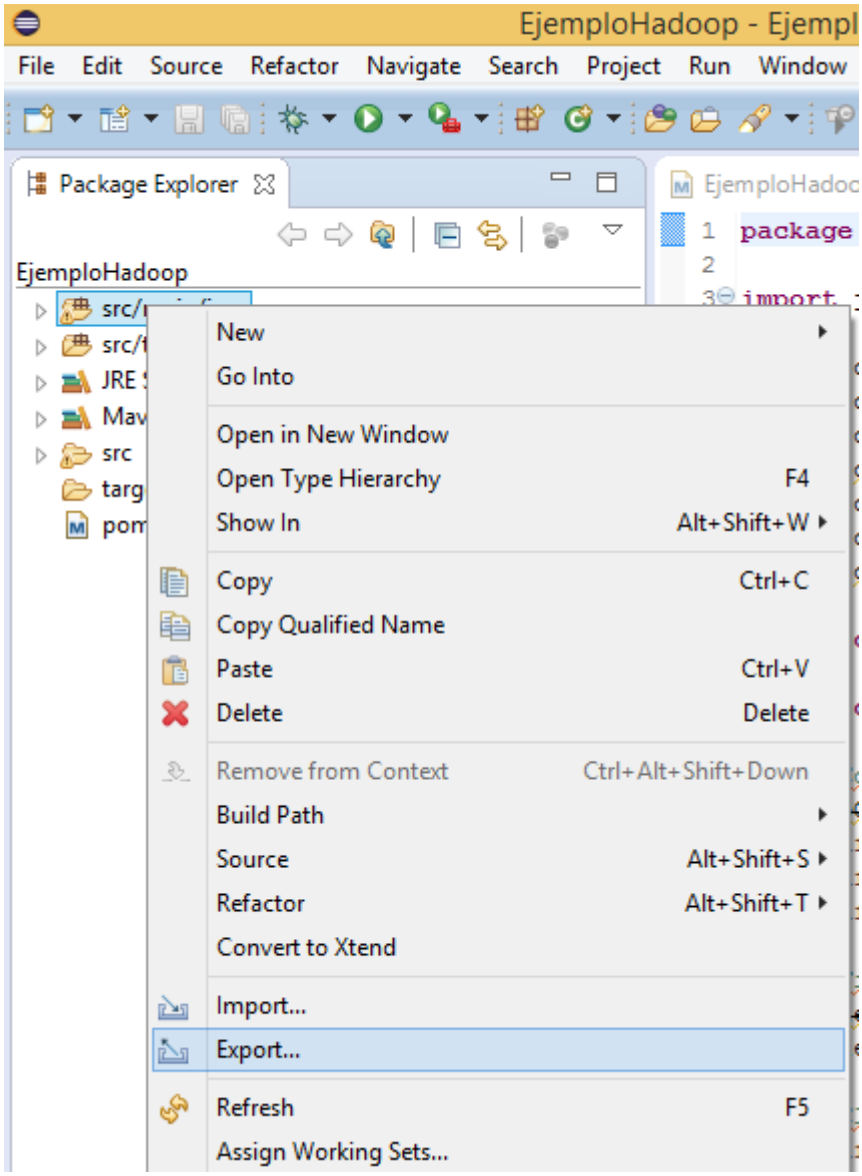


Automáticamente Maven descargara las librerías necesarias:

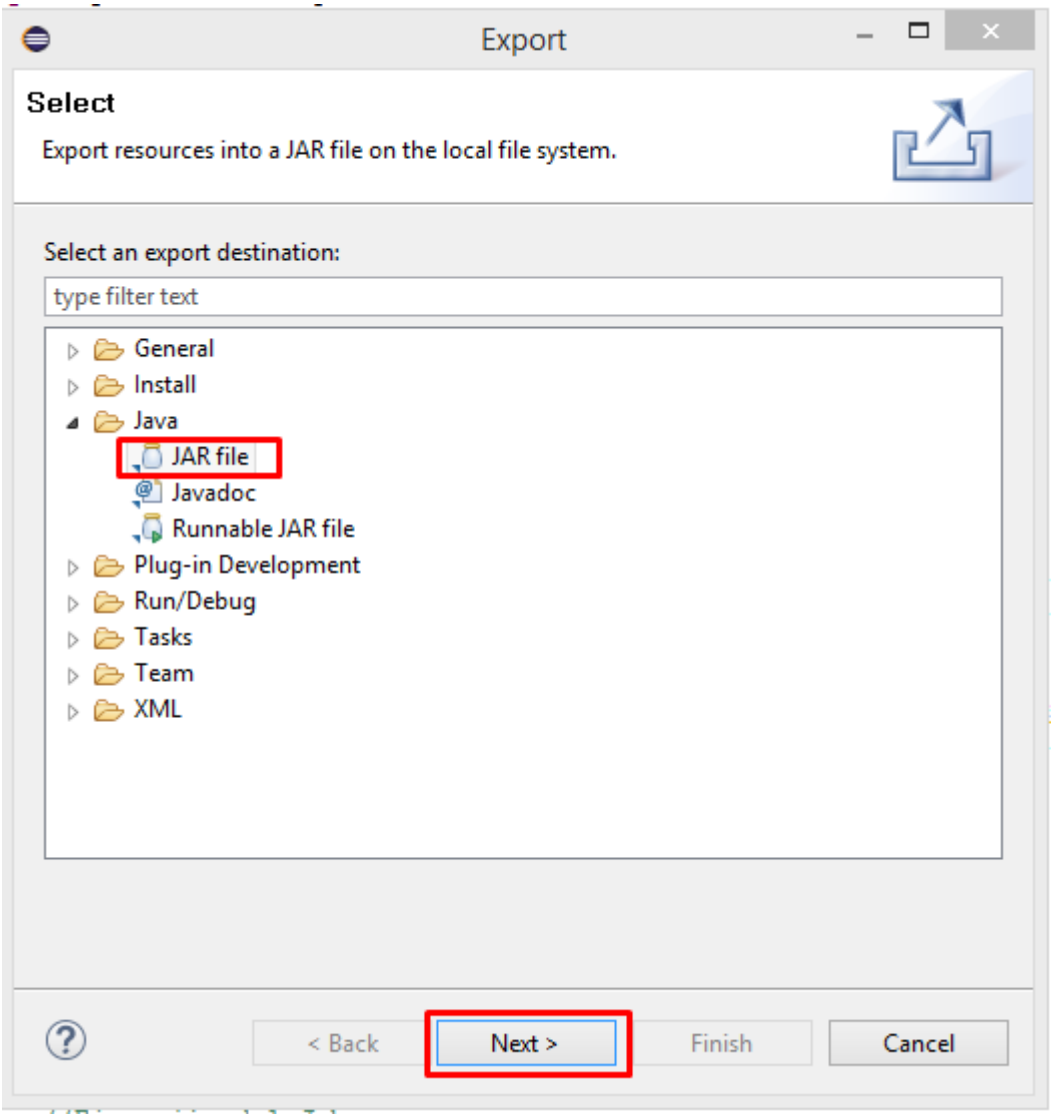


Teniendo esto procedemos a generar él .jar:

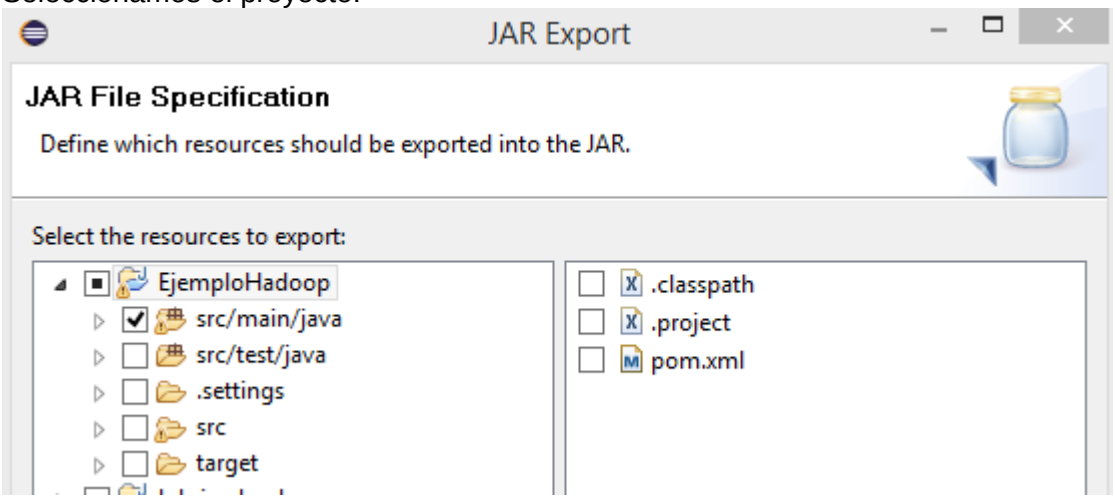
Alan Reyes Bacusoy



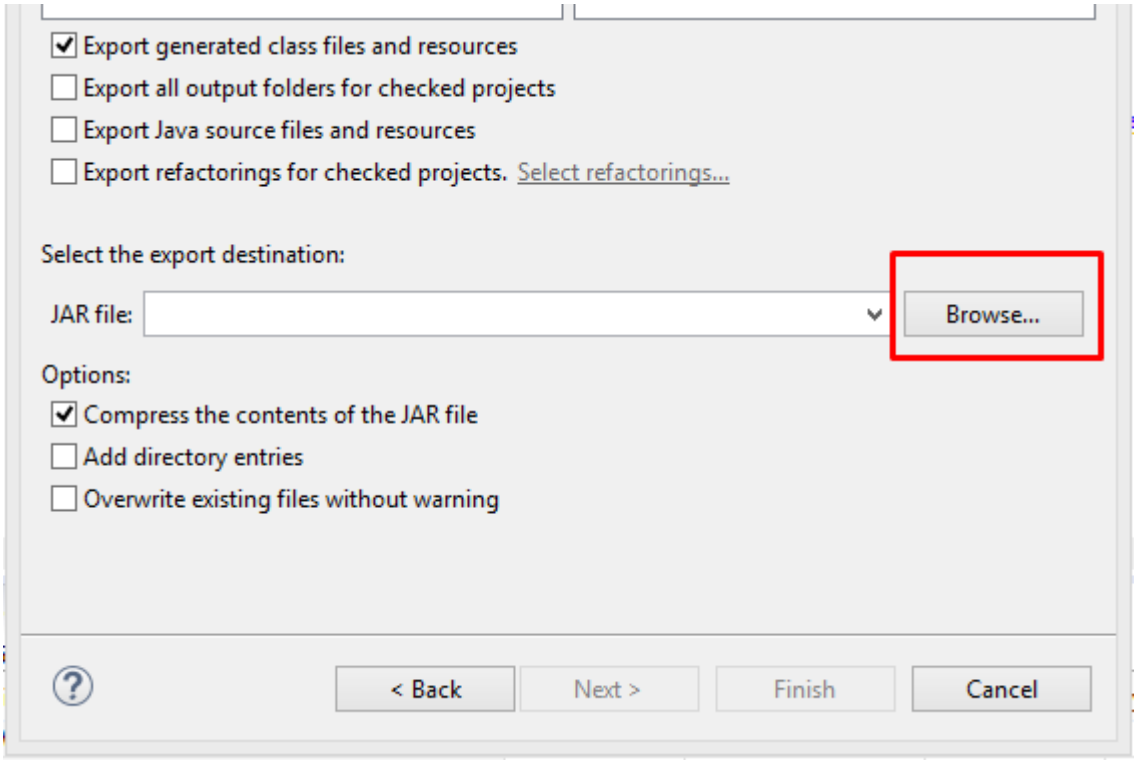
Alan Reyes Bacusoy



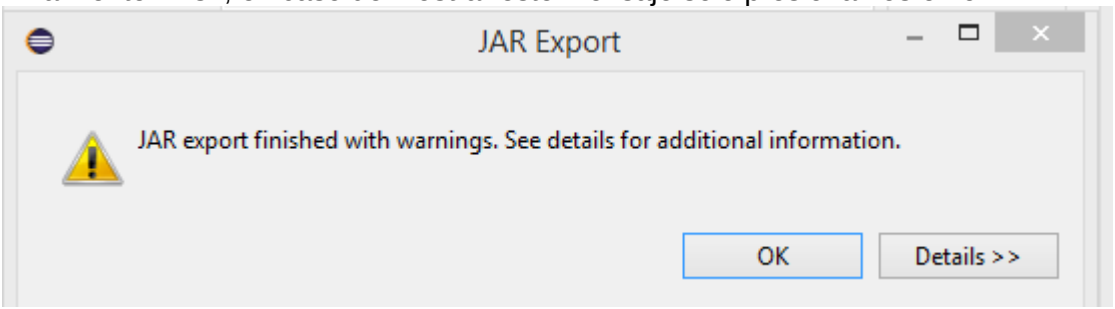
Seleccionamos el proyecto:



Buscamos la ruta en el cual queremos que se cree:



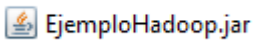
Finalmente finish, en caso de mostrar este mensaje solo presionamos en ok:



NOTA: recordemos que maven antes de generar el jar hace pruebas unitarias las cuales dan error ya que el proyecto tiene rutas preestablecidas las cuales deberán ser cambiadas en caso de generarse algún error el momento de ejecutar el .jar desde nuestro ubuntu.

Paso 2

Una vez generado el jar lo copiaremos a nuestra máquina virtual.



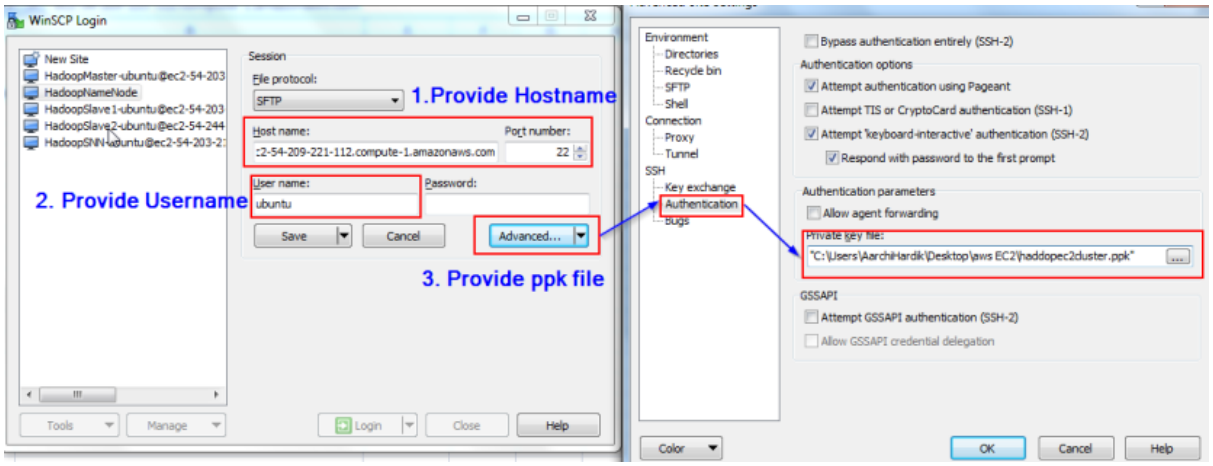
En caso de ser AWS se puede copiar el archivo como lo indica el paso 3, caso contrario avanzar hasta el paso 4:

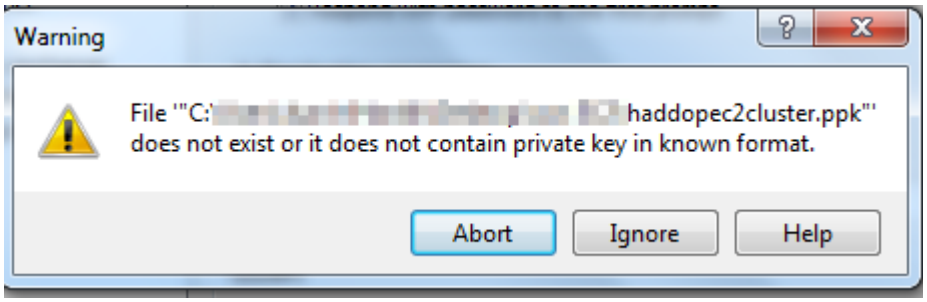
Paso 3

Debemos descargar e instalar el programa WinSCP

https://winscp.net/eng/download.php?__hstc=245485531.27469babbb1dfad919634764a629009f.1475193600064.1475193600066.1475193600067.2&__hssc=245485531.1.1475193600067&__hsfp=1773666937

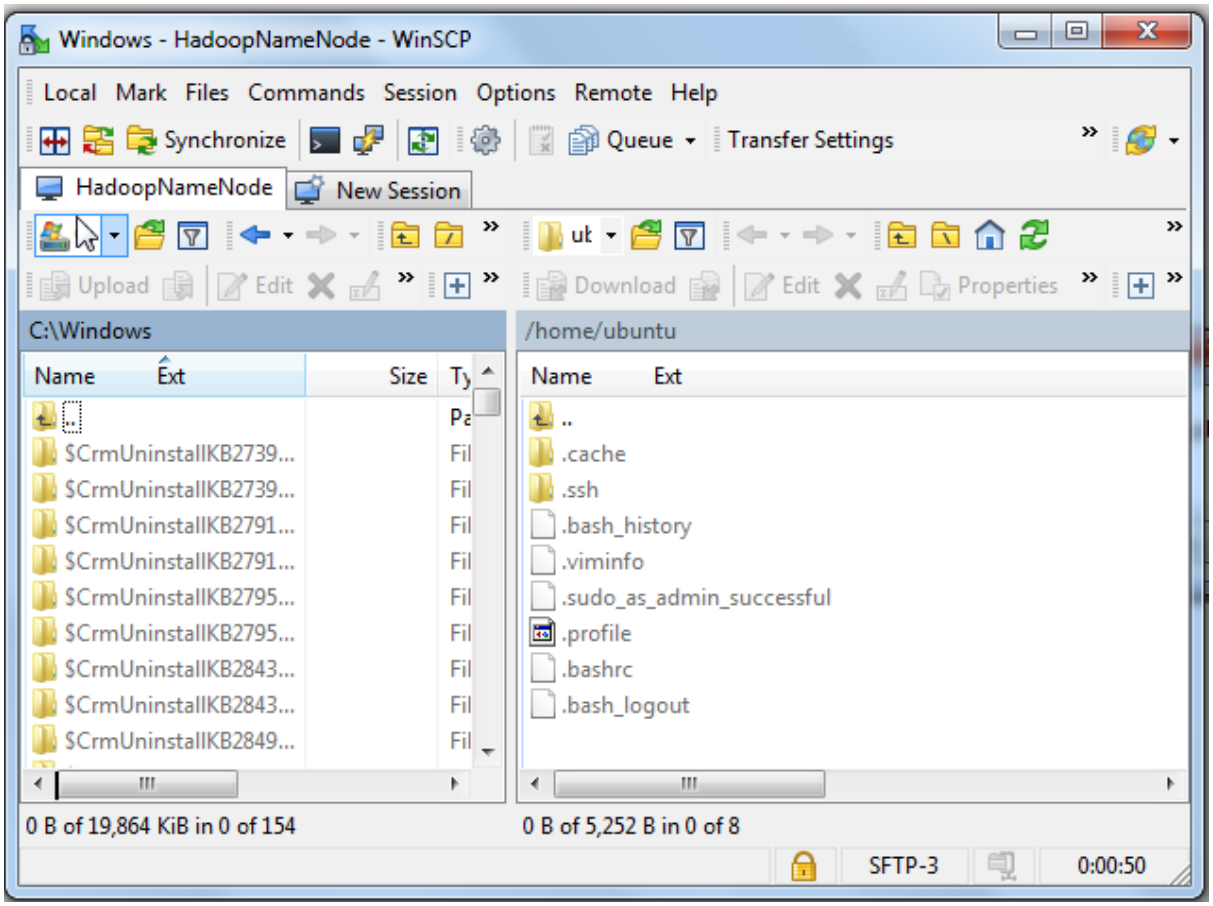
Para transferir archivos de forma segura desde su máquina Windows a Amazon EC2, WinSCP es una práctica utilidad.
Proporcione el nombre de host, el nombre de usuario y el archivo de clave privada y guarde su configuración e inicio de sesión





Si ve el error anterior, simplemente ignóralo y, al iniciar sesión con éxito, verá el sistema de archivos Unix de un usuario / home / ubuntu conectado a su máquina Amazon EC2 Ubuntu.

Alan Reyes Bacusoy



Suba el archivo .pem al equipo. Se usará cuando se conecte a nodos esclavos durante los demonios de inicio de hadoop.

Paso 4

Colocamos nuestro fichero en el sistema de ficheros de Hadoop:

bin/hadoop dfs -put /home/hduser/terremotos terremotos
bin/hadoop dfs -ls /user/hduser/terremotos

Ejecutaremos el jar: "java -jar EjemploHadoop.jar"

Y así obtenemos la lista de terremotos en Galicia a lo largo de la historia reciente:

hduser@ubuntu:/usr/local/hadoop$ bin/hadoop dfs -cat /user/hduser/terremotosOutput/part-00000

Coruña 240
Costa gallega 252
Lugo 1276
OTROS 56518
Ourense 519

Alan Reyes Bacusoy

Pontevedra 249

Si queremos volver a lanzar el jar, podemos borrar el directorio de salida con "bin/hadoop dfs -rmr /user/hduser/terremotosOutput"

```
Deleted hdfs://localhost:9000/user/hduser/terremotosOutput
hduser@ubuntu:/usr/local/hadoop$ java -jar /home/hduser/Terremotos.jar
13/08/18 12:10:35 WARN mapred.JobClient: Use GenericOptionsParser for parsing the arguments. Applications should implement Tool for the same.
13/08/18 12:10:36 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
13/08/18 12:10:36 WARN snappy.LoadSnappy: Snappy native library not loaded
13/08/18 12:10:36 INFO mapred.FileInputFormat: Total input paths to process : 1
13/08/18 12:10:36 INFO mapred.JobClient: Running job: job_201308181031_0010
13/08/18 12:10:37 INFO mapred.JobClient:  map 0% reduce 0%
13/08/18 12:10:45 INFO mapred.JobClient:  map 100% reduce 0%
13/08/18 12:10:52 INFO mapred.JobClient:  map 100% reduce 33%
13/08/18 12:10:54 INFO mapred.JobClient:  map 100% reduce 100%
13/08/18 12:10:55 INFO mapred.JobClient: Job complete: job_201308181031_0010
13/08/18 12:10:55 INFO mapred.JobClient: Counters: 30
13/08/18 12:10:55 INFO mapred.JobClient:   Job Counters
13/08/18 12:10:55 INFO mapred.JobClient:     Launched reduce tasks=1
13/08/18 12:10:55 INFO mapred.JobClient:     SLOTS_MILLIS_MAPS=11762
13/08/18 12:10:55 INFO mapred.JobClient:     Total time spent by all reduces waiting after reserving slots (ms)=0
13/08/18 12:10:55 INFO mapred.JobClient:     Total time spent by all maps waiting after reserving slots (ms)=0
13/08/18 12:10:55 INFO mapred.JobClient:     Launched map tasks=2
13/08/18 12:10:55 INFO mapred.JobClient:     Data-local map tasks=2
13/08/18 12:10:55 INFO mapred.JobClient:     SLOTS_MILLIS_REDUCES=9528
13/08/18 12:10:55 INFO mapred.JobClient:   File Input Format Counters
13/08/18 12:10:55 INFO mapred.JobClient:     Bytes Read=5143052
13/08/18 12:10:55 INFO mapred.JobClient:   File Output Format Counters
13/08/18 12:10:55 INFO mapred.JobClient:     Bytes Written=79
13/08/18 12:10:55 INFO mapred.JobClient:   FileSystemCounters
13/08/18 12:10:55 INFO mapred.JobClient:     FILE_BYTES_READ=712157
13/08/18 12:10:55 INFO mapred.JobClient:     HDFS_BYTES_READ=5143288
13/08/18 12:10:55 INFO mapred.JobClient:     FILE_BYTES_WRITTEN=1586822
13/08/18 12:10:55 INFO mapred.JobClient:     HDFS_BYTES_WRITTEN=79
13/08/18 12:10:55 INFO mapred.JobClient:   Map-Reduce Framework
13/08/18 12:10:55 INFO mapred.JobClient:     Map output materialized bytes=712163
13/08/18 12:10:55 INFO mapred.JobClient:     Map input records=59054
13/08/18 12:10:55 INFO mapred.JobClient:     Reduce shuffle bytes=712163
13/08/18 12:10:55 INFO mapred.JobClient:     Spilled Records=118108
13/08/18 12:10:55 INFO mapred.JobClient:     Map output bytes=594043
13/08/18 12:10:55 INFO mapred.JobClient:     Total committed heap usage (bytes)=476053504
13/08/18 12:10:55 INFO mapred.JobClient:     CPU time spent (ms)=5220
13/08/18 12:10:55 INFO mapred.JobClient:     Map input bytes=5141528
13/08/18 12:10:55 INFO mapred.JobClient:     SPLIT_RAW_BYTES=236
13/08/18 12:10:55 INFO mapred.JobClient:     Combine input records=0
13/08/18 12:10:55 INFO mapred.JobClient:     Reduce input records=59054
13/08/18 12:10:55 INFO mapred.JobClient:     Reduce input groups=6
13/08/18 12:10:55 INFO mapred.JobClient:     Combine output records=0
13/08/18 12:10:55 INFO mapred.JobClient:     Physical memory (bytes) snapshot=495173632
13/08/18 12:10:55 INFO mapred.JobClient:     Reduce output records=6
13/08/18 12:10:55 INFO mapred.JobClient:     Virtual memory (bytes) snapshot=2922844160
13/08/18 12:10:55 INFO mapred.JobClient:     Map output records=59054
hduser@ubuntu:/usr/local/hadoop$ bin/hadoop dfs -cat /user/hduser/terremotosOutput/part-00000
Coruña  240
Costa gallega   252
Lugo    1276
OTROS   56518
Ourense 519
Pontevedra      249
```

Bibliografía

https://www.adictosaltrabajo.com/tutoriales/mapreduce-basic/
http://papidal.blogspot.com/2013/08/big-data-configurar-hadoop-y-ejemplo-de.html
https://dzone.com/articles/how-set-multi-node-hadoop
https://www.novixys.com/blog/setup-apache-hadoop-cluster-aws-ec2/#AWS_EC2_Startup
https://sagarruchandani.wordpress.com/2015/08/01/hadoop-setting-up-hadoop-2-6-0-single-node-on-aws-ec2-ubuntu-ami/
http://www.cs.cityu.edu.hk/~heleicui2/doc/Setup-Hadoop-2.7.3-(single-node)-on-AWS-EC2-Ubuntu-AMI.pdf
https://letsdobigdata.wordpress.com/2014/01/13/setting-up-hadoop-1-2-1-multi-node-cluster-on-amazon-ec2-part-2/