**Maven: supervivencia -**

**Eclipse /VSCode howto Java Maven en consola y en Eclipse**

Es necesario ver los 9 vídeos del tutorial. Fundamentales los 6 primeros.

[Nota si estas cursando el módulo de programación: estos contenidos -todo lo relacionado con Maven- se evalúa en como metodologías de la programación.]

**Maven Tutorial for Beginners - VIDEOS:**

https://www.youtube.com/watch?v=HBXxBJ-7LFw&index=1&

list=PLS1QulWo1RIaaQ3mAU9Nj4rqfwbAv3wIZ

* Maven Tutorial for Beginners 2 - How to Install and Setup Maven:

https://www.youtube.com/watch?v=3ODSQ0EpoQI&index=2&

list=PLS1QulWo1RIaaQ3mAU9Nj4rqfwbAv3wIZ

* Sólo en caso de que sea necesario: **Installing the Java JDK Software and Setting JAVA\_HOME on Windows** :

https://www.youtube.com/watch?v=Hjs6L...

Paso a paso y comandos en:

http://www.codebind.com/maven/install-maven-windows/

* Lanzarlo desde **consola** y configurar proyecto: **Creating First Maven Project 3**

https://youtu.be/MHrUzW-4XzI

* ==>> Maven Tutorial for Beginners 4 - **Creating Maven project using Eclipse IDE + dependencia JUnit test de Unidad + Understanding pom.xml**

https://www.youtube.com/watch?v=TpPEgCm65CE

* Maven Tutorial for Beginners 5 - How to create a jar file with Maven

https://www.youtube.com/watch?v=vGtGxKZQ-l8

**Maven lifecycle**

https://maven.apache.org/guides/introduction/introduction-to-the-lifecycle.html

**Maven en 5 minutos:**

https://maven.apache.org/guides/getting-started/maven-in-five-minutes.html

**Maven Getting Started Guide**

https://maven.apache.org/guides/getting-started/index.html

**Maven Repository**

https://mvnrepository.com/

* la librería de assertj

https://mvnrepository.com/artifact/org.assertj/assertj-core-java8/1.0.0m1

* junit

https://mvnrepository.com/artifact/junit/junit/4.12

**JAR**

https://en.wikipedia.org/wiki/JAR\_(file\_format)

https://en.wikipedia.org/wiki/Java\_class\_file

**What is Hamcrest?**

https://www.captechconsulting.com/blogs/What-is-Hamcrest

https://code.google.com/archive/p/hamcrest/wikis/Tutorial.wiki

-----------------––––––––––––––––------------–––––––––––––----–––––---------

**Maven en VSCode**

Lee cómo se instala y cómo se procede con esta extensión para VSCode aquí: https://marketplace.visualstudio.com/items?itemName=vscjava.vscode-maven

**Instalar la extensión**

En la pestaña extensiones busca la extensión **Maven Project Explorer**: https://marketplace.visualstudio.com/items?itemName=vscjava.vscode-maven

**Crear la carpeta del proyecto.**

O no porque el archetype la crea por nosotros.

**Update maven archetype Catalog**

Abrir la consola:

Ctlr + Shift + P

Ejecutar el comando:

Maven: Update maven archetype Catalog

**elegir el Archetipo**

Generate from Maven Archetype.

The extension loads archetypes listed in local/remote catelog . After selection, the extension fires mvn archetype:generate -D.. . in terminal.

Ejecutar en consola el comando:

Maven: Generate from Maven Achetype

Seleccionar el arquetipo e introducir la info:

maven-archetype-quickstart

Introducir toda esta información a cada pregunta en el terminal (qué significa cada uno de estos parámetros está explicado en el vídeo anterior Creating First Maven Project 3):

groupID: org.mvpigs => unique across organization

artifactID: bicipalma => nombre del proyecto

version: 1.0-SNAPSHOT => es una versión de desarrollo; así se llamará el .jar package: org.mvpigs.bicipalma

Y:: para confirmar la anterior configuración: has de revisarla

**Tree**

Ya tenemos el árbol de Maven en el explorador con las dos primeras clases creadas, una para desarrollo otra para testing:

src > main > java > org > mvpigs > bicipalma >> app.java

test > java > org > mvpigs > bicipalma > appTest.java

**POM.XML**

Revisamos el POM.xml

Chequeamos que existe la dependencia a JUnit

<!-- https://mvnrepository.com/artifact/junit/junit -->

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.12</version>

<scope>test</scope>

</dependency>

Sino, hay que ir a los repositorios de Maven y añadirla:

https://mvnrepository.com/artifact/junit/junit

Como queremos trabajar con Java 8, chequeamos o incluímos esta propiedad, antes de las dependencias:

<properties>

<maven.compiler.source>1.8</maven.compiler.source>

<maven.compiler.target>1.8</maven.compiler.target>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

</properties>

----------------–––––––––––––––-------------------------------------------------------**JAVA en VSCode**

En la pestaña extensiones busca e instala las extensiones:

* + Language support for Java for Visual Studio Code
  + Debugger for Java
  + Java Test Runner
  + Java Extension Pack

Tendrás que recargar VSCode (MS style).

**Ejecutar código**

Crea un proyecto Maven del archetipo quickstart:

maven-archetype-quickstart

o abre una carpeta donde tengas algún proyecto Java.

**LAUNCH.JSON**

Vete a la pestaña *Debugging*. Agrega la configuración launch.json: java haciendo click en abrir launch.json (la ruedecita de mecanismo con el punto naranja llamando tu atención).

Debemos indicar en el launch.json cuál es la clase que contiene la función main . En nuestro caso, la clase App.java contiene la función main y está dentro del paquete ejemplo :

"mainClass": "ejemplo.App",

**Run**

Observa la salida en el terminal de VSCode.

**Realiza el ciclo de vida de Maven:**

Clic con botón derecho sobre el nombre de un proyecto maven en la vista *Explorador > Maven Projects*.

Selecciona en el menú emergente el comando que quieras ejecutar:

validate

validate the project is correct and all necessary information is available compile

compile the source code of the project

test

test the compiled source code using a suitable unit testing framework. These tests should not require the code be packaged or deployed

package

take the compiled code and package it in its distributable format, such as a JAR.

verify

run any checks on results of integration tests to ensure quality criteria are met

install

install the package into the local repository, for use as a dependency in other projects locally

deploy

done in the build environment, copies the final package to the remote repository for sharing with other developers and projects.

**GIT**

Vete a la vista *Control de código fuente* y pon en seguimiento el directorio que quieras del proyecto.

Es importante que configures el fichero . gitignore para que Git **no** realice un seguimiento de los directorios y ficheros de configuración del proyecto Maven, de Eclipse o de VSCode (o de tu RAD /IDE particular).

En github encontrarás ejemplos como este:

https://gist.github.com/dedunumax/54e82214715e35439227