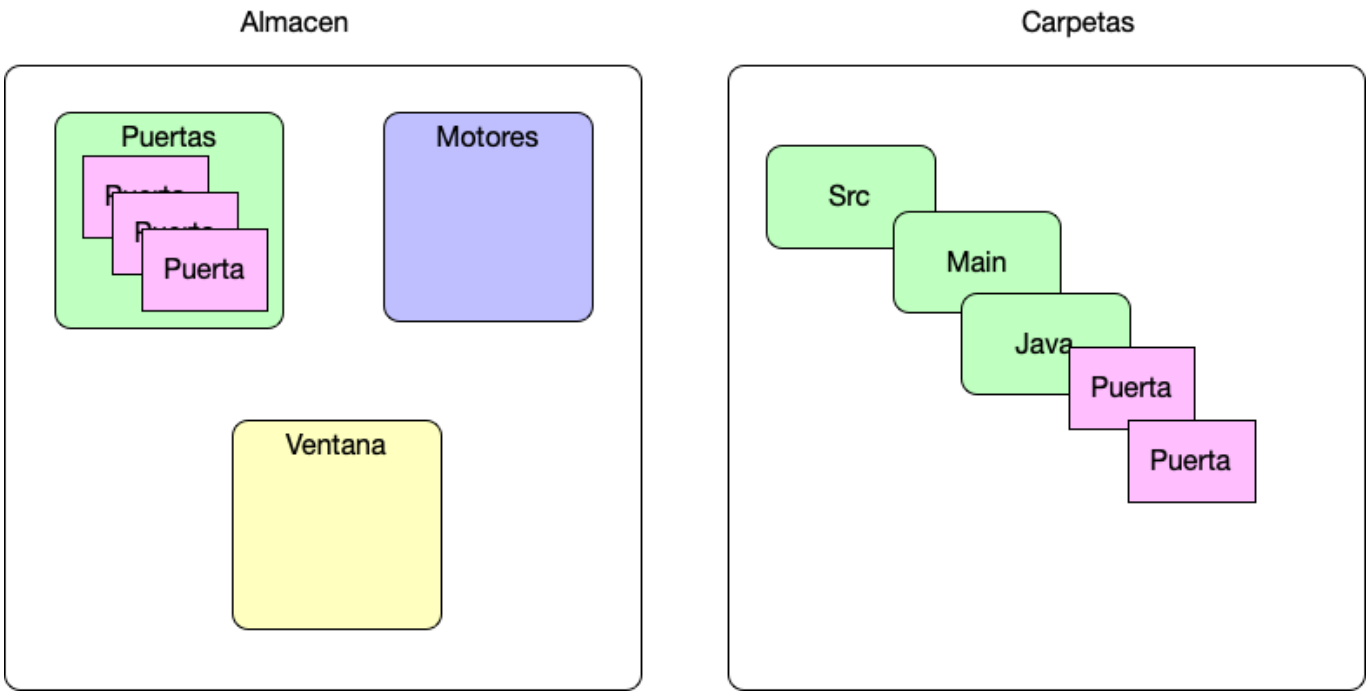
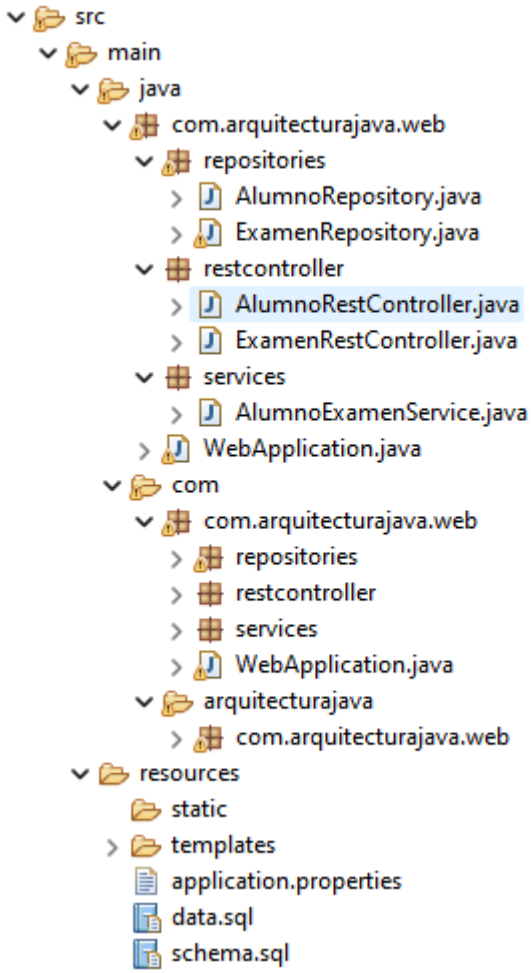
¿Qué es Maven? . Esta es quizás la pregunta mas común que un desarrollador Java Junior hace a un desarrollador Java Senior. El conjunto de respuestas que he oido es infinita . Maven es un gestor de proyectos Java , Maven es una herramienta para compilar código Java. Maven es un empaquetador , Maven no se muy bien lo que es, No uses Maven que es un lio. Maven es obligatorio… Maven …..etc. Vamos a explicar de una forma sencilla para que sirve esta Herramienta.

Maven : La fabrica de coches

Para ello vamos a usar un simil a la hora de acercarnos al concepto . Supongamos que nosotros queremos que nos fabriquen un coche . Se trata de un deseo sencillo pero … un poco caro. Ya que un coche no es algo sencillo de construir sino algo COMPLEJO. Para ello tendremos que acercarnos a la fabrica de coches y pedir el coche. Si hablamos con los encargados de la fabrica y les preguntamos como nos van a fabricar el coche . Nos dirán que todo empieza por el almacén en donde se almacenan las piezas con las cual el coche se monta (Puertas, Motor, Ruedas etc). Cada zona del almacen se encarga de guardar un tipo de elemento. Las puertas pueden ser de modelos diversos, o los motores también pero cada uno va colocado en su parte del almacen.



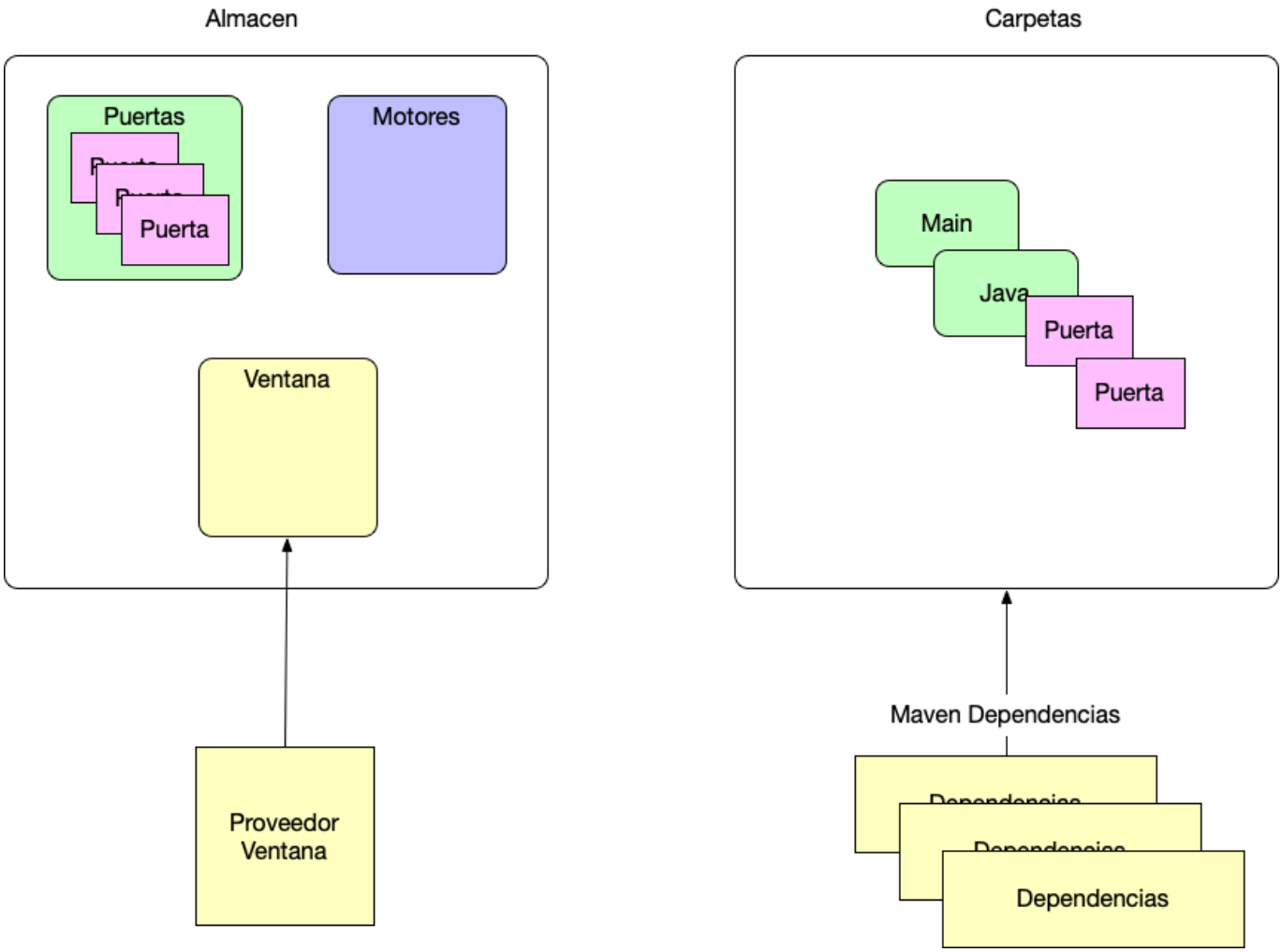
Eso es lo mismo que ocurre con el software nuestras clases y nuestros componentes .Tienen que ir ubicados en una serie de carpetas “standards” y Maven nos aporta ese estructura básica de carpetas para ubicar las piezas fundamentales para construir nuestro software. De igual manera que una fabrica aporta un almacén para colocar las piezas. La carpeta src/main/java es donde van los fuentes por ejemplo:



src/main/resources donde van recursos adicionales:

Gestión de Dependencias

Si preguntamos al fabricante de Coches si todas las puertas , motores etc los Fabrican ellos directamente te dirá que algunos sí pero que otros los subcontratan a proveedores de tal forma que vienen ya previamente montados y se almacenan hasta que sean necesarios en el montaje del coche.

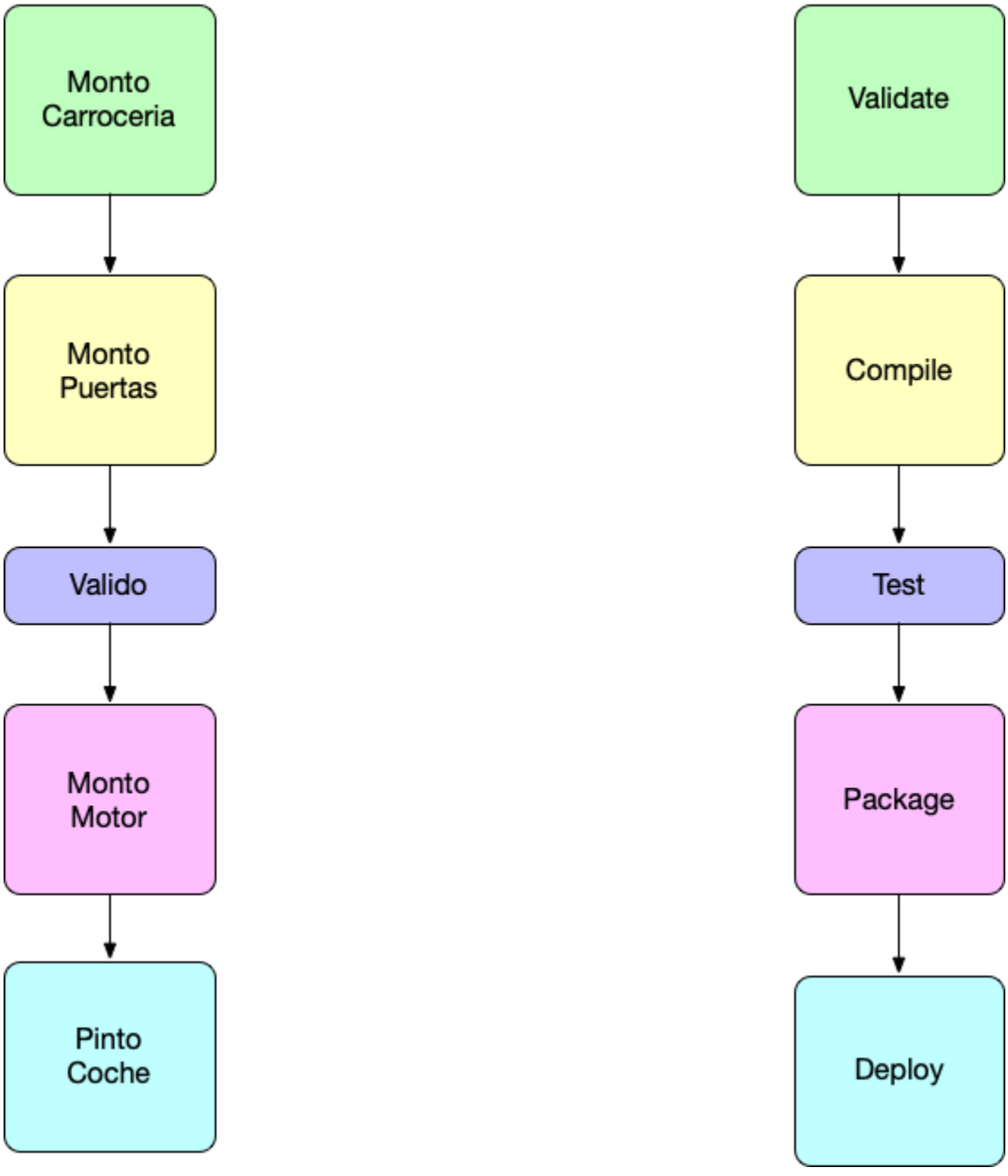


Eso es lo que hace Maven cuando incluye dependencias simplemente es comprar a un proveedor en este caso de software unas piezas que necesitamos para construir nuestro Programa.

```
<dependencies>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-data-
jpa</artifactId>
		</dependency>
		<dependency>
			<groupId>org.springframework.cloud</groupId>
```

```
                        <artifactId>spring-cloud-starter-netflix-
eureka-client</artifactId>
                </dependency>
                <dependency>
                        <groupId>org.springframework.boot</groupId>
                        <artifactId>spring-boot-starter-
thymeleaf</artifactId>
                </dependency>
</dependencies>
```

Una vez tenemos todas las piezas para construir nuestro coche tenemos que ponerlas en varias lineas de montaje que según avanzan van montando el coche y probando que todo funciona bien.

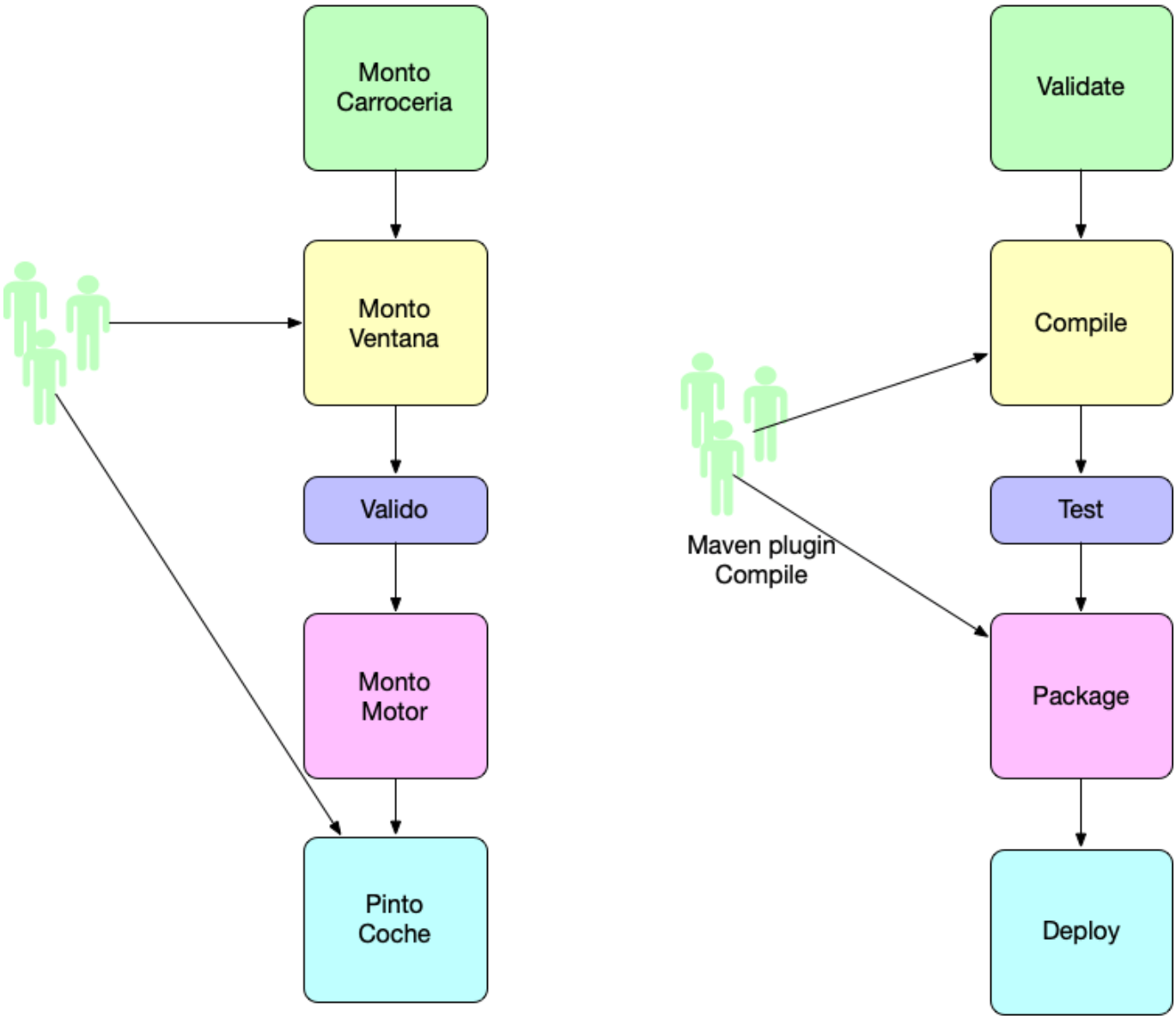


LifeCycle

Ese es el ciclo de vida por el que las piezas del coche pasan hasta tener el coche pintado y poderlo entregar .En Maven sucede lo mismo con nuestro software. Pasa por un ciclo de vida o LifeCycle de Maven que se encarga de validar, compilar , testear y desplegar nuestro código en un entorno determinado.

Plugins y equipos de trabajo

Muchas veces en este ciclo de vida que tenemos a la hora de montar el coche se usan equipos de trabajo que operan en el ciclo de vida. Es decir un grupo de operarios que por ejemplo instala las ventanas y que quizás más adelante pinta el coche.

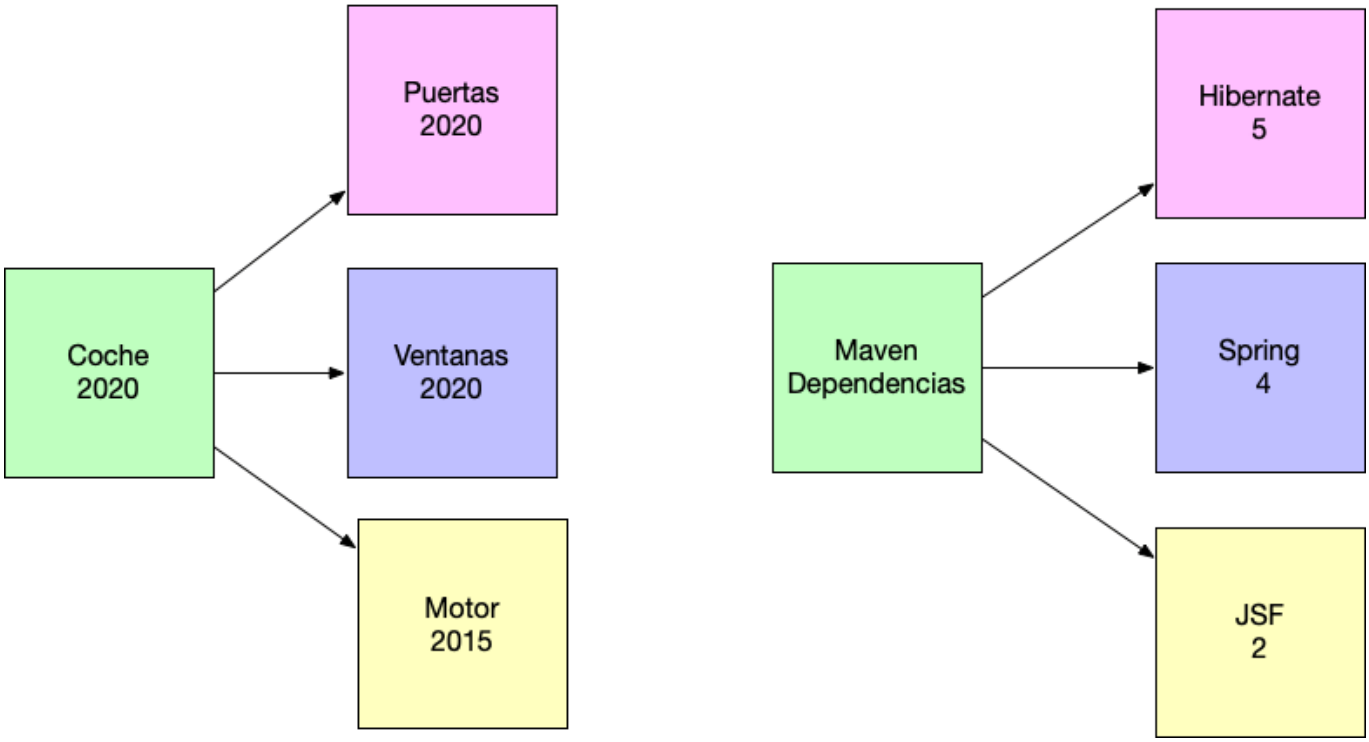


Ese es el concepto de Maven Plugin . Un Plugin es una pieza de software que opera sobre el ciclo de vida de construcción de nuestro software realizando diferentes tareas y que puede

ser parametrizado.

Coche , Maven y Versionado

Muchas veces la fabrica va evolucionando el Coche con el paso de los años y pasamos de un Toyota Yaris de 2010 a un Toyota Yaris de 2020 . Para ello se ve obligado a actualizar la versión de todas las piezas o de una gran mayoría de ellas



De la misma forma funciona Maven ya que puede necesitar diferentes dependencias en diferentes versiones para construir el software que queremos construir.

Spring boot y Maven

Vamos a ver un ejemplo clásico usando un fichero de Spring Boot:

```
<groupId>com.arquitecturajava</groupId>
<artifactId>spring-boot-data-web</artifactId>
```

```
<version>0.0.1-SNAPSHOT</version>
<packaging>jar</packaging>

<name>spring-boot-data-web</name>
<description>Aplicación Spring Boot con Spring Web y Spring Data
JPA</description>

<parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>3.1.4</version>
    <relativePath/>
</parent>

<properties>
    <java.version>17</java.version>
</properties>

<dependencies>
    <!-- Spring Boot Starter Web -->
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-web</artifactId>
    </dependency>

    <!-- Spring Boot Starter Data JPA -->
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-data-jpa</artifactId>
    </dependency>
```

```
    <!-- H2 Database  -->
    <dependency>
        <groupId>com.h2database</groupId>
        <artifactId>h2</artifactId>
        <scope>runtime</scope>
    </dependency>



    <!-- Spring Boot Starter Test -->
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-test</artifactId>
        <scope>test</scope>
    </dependency>
</dependencies>

<build>
    <plugins>
        <!-- Spring Boot Maven Plugin -->
        <plugin>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-maven-plugin</artifactId>
        </plugin>
    </plugins>
</build>
```

El archivo `pom.xml` es la configuración principal en un proyecto Maven. Aquí te explico de manera sencilla el contenido:

1. Encabezado del proyecto:

- groupId: Identifica de manera única el grupo u organización del proyecto. En este caso, usamos com.arquitecturajava, que refleja tu dominio.
- artifactId: Nombre del proyecto o módulo. Aquí es spring-boot-data-web.
- version: Versión del proyecto. En este caso, 0.0.1-SNAPSHOT indica que está en una fase de desarrollo.
  2. Parent:

- Indica que el proyecto hereda de spring-boot-starter-parent, una configuración básica proporcionada por Spring Boot que facilita la gestión de dependencias y versiones.
  3. Propiedades:

- java.version: Establece la versión de Java a usar. En este caso, la versión es 17.
  4. Dependencias: Estas son las bibliotecas que el proyecto necesita:

- spring-boot-starter-web: Incluye todo lo necesario para crear aplicaciones web usando Spring MVC.
- spring-boot-starter-data-jpa: Proporciona integración con JPA para interactuar con bases de datos relacionales.
- H2 database: Es una base de datos en memoria para hacer pruebas rápidas o usar en pequeñas aplicaciones.
- spring-boot-starter-test: Incluye herramientas para realizar pruebas unitarias y de integración en el proyecto.
  5. Build (Construcción):

- spring-boot-maven-plugin: Este plugin permite ejecutar la aplicación directamente desde Maven con mvn spring-boot y empaquetar la aplicación en un archivo jar.
  En resumen, este pom.xml define las dependencias y configuraciones necesarias para una aplicación Spring Boot que incluye funcionalidades web y acceso a bases de datos. Además, permite ejecutar la aplicación fácilmente y empaquetarla para producción.

Conclusión : ¿Que es Maven?

No es ni mas ni menos que una herramienta para fabricar software de forma Standard y como buena fabrica no es algo sencillo de entender a nivel global en un primer vistazo.

Otros artículos relacionados

- ¿Qué es un Maven Property y como se utiliza?
- Curso Maven y buenas prácticas
- ¿Que es un Maven Goal ?
- Curso Maven