Clase 19/3

#paw

Dividir aplicación en módulos ➜ obliga a programar con interfaces (facilita IoControl) y que falle en compilación

---

quickstart: grpId: ar.edu.itba.paw
'artifactId': persistance
…

---

man archetype:generate

'groupId': ar.edu.itba.paw
…

Nunca WEBAPP debería hablar con PERSISTENCE. Tampoco habla con la implementación de SERVICES, sino con las interfaces de SERVICES (lo mismo para todas las capas). ➜ Best practica: evitar errores de runtime ➜ error de compilación

Scope en mvn

Dependencias:

COMPILE: la necesito en tiempo de ejecución. disponible a la hora de compilar y cuando empaqueto.

PROVIDED: lo quiero a la hora de compilar, pero no lo quiero dentro del paquete (me lo provee jetty, tomcat, etc.) (va a haber disponible una implementación).

RUNTIME: lo quiero dentro de mi paquete, pero no existe cuando compilo. ESTO me permite programar con interfaces.

TEST: una dependencia TEST solo la quiero disponible cuando compilo o corro, no cuando empaqueto ni compilar source main.

Para Eclipse solo existe Compile y Test.

WEBAPP necesita como dependencia a SERVICES y a interfaces (porque las voy a implementar). Interfaces disponibles solo en tiempo de compile. SERVICES solo en runtime.

SERVICES necesita como dependencia a PERSISTENCE y a interfaces (porque las voy a implementar). Interfaces disponibles solo en tiempo de compile. PERSISTENCE solo en Runtime.

PERSISTENCE necesita como dependencia interfaces (porque las voy a implementar)

interfaces necesita como dependencia MODEL

META-INF genera metadata del JAR

Qué tiene el webapp.war? otros jar.

Al padre le agrego como dependencies spring webmvc y spring context.

DAO [Data Access Object]

Unico responsable de crear instancias de mi modelo.

Genera en orden (resuelve grado de dependencias en tiempo de ejecución):

Controller ➜ Service ➜ DAO ➜ DataSource