**INYECCIÓN DE DEPENDENCIAS EN SPRINGBOOT**

* Para realizar inyección de dependencias, en este ejemplo tenemos 2 interfaces (IPersonaRepo, IPersonaService) y por ende dos clases que las implementan (PersonaRepoImpl, PersonaServiceImpl).
* En las implementaciones se define un método registrar, solo que el lado de el servicio va a tener inyectada la dependencia que dirige al lado del Repositorio.

**public** **class** PersonaServiceImpl **implements** IPersonaService{

**private** **static** Logger *LOG*= LoggerFactory.*getLogger*(PersonaServiceImpl.**class**);

**private** IPersonaRepo repo;

**public** **void** registrar(String nombre) {

repo= new PersonaRepoImpl();

repo.registrar(nombre);

}

}

* Allí vemos que se tiene la instancia (repo) a la interface del lado del repositorio. Este lado para implementar el método registrar lo que hace es invocar al registrar del lado del repositorio que es quien “tiene los datos”.
* Como vemos debemos crear la instancia de la clase que va a implementar esa interface que se esta creando, de esta manera podríamos invocar métodos de la clase como tal.
* De igual manera se haría desde la clase main():

**private** **static** Logger *LOG*= LoggerFactory.*getLogger*(DemoConsolaApplication.**class**);

**private** IPersonaService service;

**public** **static** **void** main(String[] args) {

SpringApplication.*run*(DemoConsolaApplication.**class**, args);

}

@Override

**public** **void** run(String... args) **throws** Exception {

// **TODO** Auto-generated method stub

service= new PersonaServiceImpl();

service.registrar("JHONATAN ZUNIGA");

}

* Solo que al hacerlo de esta manera, tendríamos que a mano instanciar cada vez que se quiera acceder al método desde un cliente distinto, por lo que se debe utilizar Estereotipos para indicar que se cree la instancia automáticamente al hacer la inyección de dependencias.
* Para esto, la clase a la que se quiere que se maneje por Spring y que además es quien accede a los datos, se le coloca el stereotype **@Repository**
* A la clase que representa la lógia de negocio, se le coloca el stereotype @Service.
* Ahora, para asignar esa instancia a la interface automáticamente, lo que se hace es que cuando se crea el objeto inyectado, se le antepone el stereotype **@Autowired,** de esta manera se hace un “= new ….” Automáticamente.

**public** **class** DemoConsolaApplication **implements** CommandLineRunner{

**private** **static** Logger *LOG*= LoggerFactory.*getLogger*(DemoConsolaApplication.**class**);

@Autowired

**private** IPersonaService service;

**public** **static** **void** main(String[] args) {

SpringApplication.*run*(DemoConsolaApplication.**class**, args);

}

@Override

**public** **void** run(String... args) **throws** Exception {

// **TODO** Auto-generated method stub

service.registrar("JHONATAN ZUNIGA");

}

}