Spring Framework

Dependency Injection – pattern

Dependency Injection containers

-Object graph of any problem implementation in OOP.

IoC

Class Laptop

{

HitachiHD obj1=new HitachiHD() // hardcoding when we use new [Tightly coupled] //needs abstraction so that in future any changes no need to change the laptop class.

}

Abstract / Interface : Harddrive

Class Laptop

{

Hardrive hd=?? //Now HD can have any hardrive of any subtype of hardrive

}

Why we need this?

Tight coupling.

Testing components. [Mock object can be created only if we have loose coupling]

Better maintainence

Spring-Boot Framework

Developed by pivotal, they have their own IDE STS build over eclipse. It has got some started templates for spring boot application.

Spring-Container

* Used for fast development of enterprise applications.
* Use of Dependency Injection

Lap1

@Component(“lap1”)

Laptops

{

private int ram;

private int id;

private String name;

private String decription;

}

Alien is assigned with laptop

Singleton object injected to alien

Aliens

{

private int id;

private int name;

private String location;

@Autowired

@Qualfier(“lap1”)

private Laptop lappy;

Aliens(Laptop lap)

{

//lap = new Laptop(); ?? this is hardcoding for creation of the object

}

}

To avoid this kind of tight coupling , injection can be used to make the component loosely coupled.

Laptop is annoted with @Component annotation.This will create the object in the spring container by default this object is singleton.

-To specify scope @Scope(value=”prototype”) needs to be mentioned.

-In the Alien either from the spring context using getbean method the component object can be fetched or using the annotation @Autowired the component can be injected.

-@Qualifier(“lap1”) can be used to fetch object by specific name\_id.