Spring Framework: Spring is an open source layer architecture or onion architecture framework.

Spring framework provided n number of layer or module to improve all types of application.

Spring modules

1. Spring core
2. Spring context
3. Spring mvc : spring MVC model centric framework
4. Spring rest api
5. Spring dao
6. Spring orm
7. Spring aop
8. Spring micro service
9. Spring cloud
10. Spring security
11. Spring testing
12. Spring boot
13. Spring integration

Spring Core and Spring Context

IOC: Inversion of Control

IOC is design pattern or programming design pattern. According to IOC in place of creating or maintaining any resources like object creation, database connection, file handling, security etc. explicitly allow to create and maintain by container. If container it maintain properly. Whenever we need any resource pull from container use it and leave it. The life of the resource taken care by container.

IOC is a concept.

Container : run time environment for type of application (JRE)

If we want to run Servlet and JSP we need web container. Web container is an engine part of Web Server ie Tomcat.

If we want to run EJB program we need EJB container. EJB container is a part of Application server.

Spring Framework provided light weighted container part of xml file and jar file.

DI : Dependency Injection

DI is an implementation of IOC. Using the DI container inject resource to programmer or developer

Types of DI

Constructor base DI

Setter base DI

To achieve constructor or setter base DI we need to do configuration

1. Using XML
2. Using annotation

**POJO** : Plain Old Java Object . it is a type class that class not to extends or implements any pre defined class except Object class. POJO is equal to JavaBean in java technologies.

**Auto wired :** Spring framework by default do the DI for primitive property implicitly with default value respective data types. If class contains complex property or user defined object that time we need to us ref attribute part property or constructor-arg tag to do DI explicitly.

With help of auto wired features we can achieve DI for complex property or user defined object without ref attribute(implicitly rather than explicit with ref attribute).

byType : spring container scan xml file. If that type bean definition present it automatically do the DI for that property.

In byType we need only one bean definition of that type.

byName : if mor than one then we need to use byName. In byName id name and reference name must be match.

@Component annotation this is class level annotation. This annotation we need to use on POJO or JavaBean class. this is generic annotation.

By default id is class name using camel naming rules. Means if class contains one word then id must be lower case. Example Address class id is address if class contains more than one word then id must follow camel naming rules. Example AddressDetails then is id addressDetails.

If class contains complex property or user defined object then we need to use @Autowired

By default @Component annotation not enable. We need to enable using

1. Using xml file
2. Using class classes with few more annotation.

@Value this annotation we can use on primitive property level to set initial value.

@Scope : this annotation we can write on class level to set scope.

@Configuration this is class level annotation. Which contains more than one configuration details. It is equal to beans.xml file.

@ComponentScan: class level annotation to scan the package.