Day 2 : 11 Sep 2024

Auto wired :

Spring framework by default do the DI for primitive property implicitly. Like int, float, char, double or string.

If class contains complex property it can be user defined object, array or list object.

Then we need to the DI for those property explicitly using property ref or constructor-arg ref attribute.

Using auto wired features we can do the DI for complex property implicitly rathe than explicitly with property ref or constructor ref attribute.

Type of auto wired

byType : spring container check type of object bean definition in xml file. If present it automatically do the DI for that particular objects. In this operation only one bean definition must be present.

byName : in byName option we can write more than one bean definition of that type. In this option reference name and bean id name must be match.

DI using annotation base

@Component : This annotation we need to use on class level. This annotation generally use on POJO or Java Bean class.

By default id name is class name using camel naming rules. Means if class contains one word then id must class name in lower case like Employee then id is employee. If class contains more than one word like EmployeeDetails then id must be employeeDetails.

@Autowired : This annotation we need to use on complex property or object. this annotation property level annotation.

By default @Component annotation is not enable. To enable this annotation we can use

1. Xml file
2. Using class with @Configuration annotation

BeanFactory is core interface which provide set of method to pull the object from container.

ApplicationContext is special interface which internally extends BeanFactory. Which also provide extra method to do the DI as well as it support annotation base DI.

@Value annotation :This annotation we can use on property level to set default value.

By default scope is singleton

@Scope : this annotation we can use on class level to set the scope. By default singleton.