1. what is IOC, Dependency injection?
2. IOC:- (Inversion of Control)

Inversion of Control (IoC) means to create instances of dependencies first and latter instance of a class (optionally injecting them through constructor), instead of creating an instance of the class first and then the class instance creating instances of dependencies.

It is used to achieve loose-coupling between Objects dependencies

IOC Containers :-

The Spring IoC container is at the core of the Spring Framework. The container will create the objects, wire them together, configure them, and manage their complete life cycle from creation till destruction. The Spring container uses dependency injection to manage the components that make up an application

There are two types of IOC containers;

They are: 1. bean factory

2. Application context

Dependency injection:-

* When two classes are tightly coupled then don’t create instance of Child class inside the parent class instead the IOC container will take care of objects creation and do setteng via constructoe (or) setter-getter injection

1. what is @Component, @Service, @Repository, @Transaction?
2. @Component:-

* @Component is an annotation that allows Spring to automatically detect our custom

beans.

* In other words, without having to write any explicit code, Spring will: Scan our application for classes annotated with @Component. Instantiate them and inject any specified dependencies into them and Inject them wherever needed

@Service:-

* @Service annotation is used in your service layer and annotates classes that perform service tasks.
* This annotation is used on a class. @Service marks a Java class that performs , performing calculations, and calling external APIs. This annotation is a specialized form of the @Component annotation intended to be used in the service layer

@Repository:-

* @Repository annotation is used to indicate that the class provides the mechanism for storage, retrieval, search, update and delete operation on objects.
* It is a specialization of the @Component annotation allowing for implementation classes to be autodetected through classpath scanning.

@Transaction :-

* The @Transactional annotation is the metadata that specifies the semantics of the transactions on a method. We have two ways to rollback a transaction: declarative and programmatic. The default rollback behavior in the declarative approach will rollback on runtime exceptions.
* Transactional annotation provides the application the ability to declaratively control transaction boundaries on CDI managed beans, as well as classes defined as managed beans by the Java EE specification, at both the class and method level where method level annotations override those at the class level.

1. What is the difference between @Component and @Bean?

|  |  |
| --- | --- |
| @Component | @Bean |
| It is a class level annaotation | It is a method level annaotation |
| Here, specific implementation is not possible | Specific implementation depending on dynamic state.we can control instance creation logic |
| @Configuration is not needed | @configuration is needed |
| It has implicit mapping | It has implicit mapping |
| @controller,@repository, and@service are specialization of @component | There are no specializations |

1. What is spring boot and what is @SpringBootApplication?
2. spring boot:-

* Spring Boot is a tool that makes developing web application and microservices with Spring Framework faster and easier through three core capabilities:

1. Autoconfiguration.
2. An opinionated approach to configuration.
3. The ability to create standalone applications

* Spring Boot aims to shorten the code length and provide you with the easiest way to develop a web application. With annotation configuration and default codes, Spring Boot shortens the time involved in developing an application.

@SpringBootApplication:-

* Spring Boot helps developers create applications that just run. Specifically, it lets you create standalone applications that run on their own, without relying on an external web server, by embedding a web server such as Tomcat into your application during the initialization process
* Spring Applicaton class is used to bootstrap and launch a Spring application from a Java main method. This class automatically creates the Applicationcontext from the classpath, scan the configuration classes and launch the application
* @SpringBootApplication annotation is used to mark a configuration class that declares one or more @Bean methods and also triggers auto-configuration and component scanning

1. How mamy types of autowire and what is default?

Types of Autowire:-

* Autowiring functionality has five modes

1. no:- It’s the default autowiring mode. It means no autowiring.
2. byName:- this mode injects the object dependency according to name of the bean. In such a case, the property and bean name should be the same. It internally calls the setter method
3. byType:- this mode injects the object dependency according to type. So it can have a different property and bean name. It internally calls the setter method.
4. constructor:- The constructor mode injects the dependency by calling the constructor of the class. It calls the constructor having a large number of parameters.
5. autodetect:- In this mode, Spring first tries to autowire by the constructor If this fails, it tries to autowire by using  bytype

Default:-

* The default mode is **'no'** i.e no auto wiring is enabled by default.
* But, if you choose to use annotation-based configuration(<context:annotation-config>), and for instance, if you are using @Autowired , then the default method of auto wiring for this is "byType"

1. what is qualifier used for?
2. There may be a situation when you create more than one bean of the same type and want to wire only one of them. In such cases, you can use the @Qualifier annotation along with @Autowired to remove the confusion by specifying which exact bean will be wired.

* The @Qualifier annotation is used to resolve the autowiring conflict, when there are multiple beans of same type
* The @Qualifier annotation can be used on any class annotated with @Component or on method annotated with @Bean .
* This annotation can also be applied on constructor arguments or method parameters.

7.what is scope of bean, what is default scope?

scope of bean:-

Bean scope decides which type of bean instance should be returned to the caller from the Spring container. Scopes are defined using @Scope annotation.

There are six types of bean scopes

1. singleton
2. prototype
3. request
4. session
5. application
6. WebSocket

default scope:-

Singleton is the default scope for a Bean, the one that will be used if nothing else is indicated. This scope implies that Spring container will create an only shared instance of the class designated by this bean, so each time the Bean is required the same object will be injected

8.what is cyclic dependency issue in spring and how to avoid it?

A. cyclic dependency issue:-

* Circular dependency in Spring happens when two or more beans require instance of each other through constructor dependency injections. For example: There is a ClassA that requires an instance of ClassB through constructor injection and ClassB requires an instance of class A through constructor injection

How to avoid it:-

* To avoid Cyclic dependencies we need to Repackage them so they are no longer mutually dependent
* We can Combine them into a single component.
* By Redesign, When you have a circular dependency, it's likely you have a design problem and the responsibilities are not well separated
* By Using Setter/Field Injection
* By Using @PostConstruct
* By Implementing ApplicationContextAware and InitializingBean

9. what is difference between application context and beanfactory and types of application context?

A. Difference between application context and beanfactory:-

|  |  |
| --- | --- |
| application context | beanfactory |
| It supports annotation based dependency injection @Autowired etc, | It does not supports annotation based dependency injection |
| application context can publish events to beans | It does not support to publish events to beans |
| application context supports for internatinalization | It does not support to intenationalization |
| Its by dedfault support Aggressive loading,application context instantiate bean when container is strated. It doesn’t wait for getbean() to be called | By default it supports Lazy loading.Beanfactory insrantiate bean when you call getBean() method |
| It allows multiple to configure multiple configuration files | It doesn’t allows multiple to configure multiple configuration files |

Types of application context:-

There are five types of applicationcontext;

* AnnotationConfigApplicationContext
* AnnotationConfigWebApplicationContext
* XmlWebApplicationContext
* FileSystemXMLApplicationContext
* ClassPathXmlApplicationContex

10.how to write constructor injection in spring?

A. We can inject the dependency by constructor.

The <constructor-arg> subelement of <bean**>** is used for constructor injection. Here we are going to inject

1. primitive and String-based values
2. Dependent object (contained object)

3.Collection values etc.