

1. What is design patterns?

ans. Design patterns are used for best practises for writing code.

ex:Factory Design, Micro design

2. Monolith vs Microservices architecture?

ans. All functionalities are developing as single page are called Monolith arc.

Developing functionalities as a small comoponents are called Microservices arc.

ex: product, cart extc..

3.What is framework?

ans. Frame work is a semideveloped software.[It provides common logic which are required for application developement.]

ex: Hibernate,structs, spring etc...

4.what is Hibernate:

ans It an ORM framework used to develop presistance layer.(CRUD)

5.what is Struts :

ans It is a web framework used to develop web layer.

6.what is Spring:

ans. By using Spring we can develop end to end.

7.Spring Advantages?

ans. 1.free and open source and light weight.

2.it is versatile(adjustable easely for other framework)

3.it is an non- invasive (it won't force to use any interface)

4.Spring works on POJO and POJI model.

8. Note : Spring is loosely coupled

9. What is inheritance and composition.

ans. inheritance: creating new class for existing class called inheritance.

Composition: creating an object of a class is called Composition.

10. Note : If you use inheritance and composition both are tightly coupled.

11. Spring core:

ans. Spring core suggests developer to follow "strategy design" pattern to develop classes, for loosely couple.

12. what is Strategy Design pattern?

ans. it comes under behavioural design pattern.

Rules:

1.Favour composition over inhertance.

2.Code to interface instead of implementation classes.

3.Code should be open for extension and closed for modification.

13.Note: Instaed of talking directly implementatin talk with interfaces.

14.What is Dependency Injection.?

ans. The process of intecting one class object into another class is called Dependency injection.

15. How many ways we can perform dependency injection.

ans. 3-wasy: 1. setter injection

2.constructor injection

3.field injection

16. what is Setter injection?

ans. The process of injection one class object into another class through setter is called setter injection.

17. what is constructor injection?

ans. The process of injection one class object into another class through constructor is called constructor injection.

18. what is field injection?

ans. The process of injection one class object into another class through constructor is called constructor injection.

19. what is IOC in spring?

ans. It is a principle which is resposible to manage and colloborate dependencies among the classes.

20. Note: In spring IOC will perform DI

21. How many ways you perform IOC?

ans we can perform in 3-ways.

- 1.XML config
- 2.Anotation
- 4.java config

-----> We should tell IOC to these are the dependent classes and these are the target classes.

22.Note: Spring supports xml, Spring boot doesn't support xml.

23.property tag represents setter injection in xml file.

24.ref represents which object you are going to be inject.

25.getBean() method, it is used for creating object.

26. Note:If you use both setter & constructor setter injection will be override.

ex: Demo d=new Demo()----->Constructor first constructor will be created here
d.setPay()----->

27.What is bean scope?

ans. Bean means how many objects should be create.

By default in spring boot bean scope is singleton(Means only one)

28.can we configure bean scope?

ans. yes, 1.singleton

2.prototype

3.request

4.session

29.Note: For prototype beans when ever we call .getBean() at that time only Object created.

30.What is Autowiring?

ans. Autowiring is used to perform DI(Dependency Injection)

In spring loc container will perform autowiring.

using ref attribute is called manual wiring, spring support autowiring.

Spring has capability to identify dependent and inject into target.'

31.What are the modes in autowiring?

ans. we have 1. byname

2. byType

3. constructor

4. no

byname: means if target class variable name matched with any bean configuration file then IOC will consider that as dependent bean. and it will inject that dependent bean object into target bean.

Note: if you use byName for autowiring it should match with variable name.

if you don't want to consider with bean name go with byType mechanism.

Note: byName will check with variable

byType will check with dataType

31. what is autowire candidate?

ans. it is used to resolve ambiguity.

32. Note: when we configure autowiring "byName" or "byType" it is performing setter injection by default.

If we want to perform autowiring thorough constructor then use constructor mode.

33. what is spring boot?

ans. it is an approach to develop spring based application.

34. what are the Spring advantages.

1.Auto configuration

2.POM starters.

3.Embedded servers.

4.Rapid development.

5.Actuators.

1. Auto configuration: Starting IOC container, Creating connection pool, creating SMTP connections, web application deployment and dependency injection

2.POM starters: Pom starters are ntg but dependencies that we use to develop our application

web starter, jdbc-starter, security-starter and mail-starter.

Boot provided we servers to run our web application those servers are called as embedded server.
tomcat(default)

5. Actuators: actuators are used to check health condition of application. Ex: how many beans created etc.

35. In spring boot what is SpringApplication.run();

ans. It contains bootstrapping log, and it is the entry point for boot application execution and run method is the responsible to create IOC

36. @SpringBootApplication annotation is equals to below three annotations?

ans. @SpringBootConfiguration

@EnableAutoConfiguration

@ComponentScan

37. For boot starter run method using "AnnotationConfigApplicationContext" class to start IOC container.

38. For Boot-web-starter run method using "WebServerApplicationContext" to start IOC container.

39. For Web-Flux starter run method using "AnnotationConfigReactiveWebServerApplicationContext" to start IOC

Note: Based on the starter it is going to be start.

40. Boot starter is used to create standalone application, boot-starter-web dependency are used to develop web application.

41. What are the runners present in spring boot?

ans. runner are getting called at end of run() method.

In spring we have 2-types of runners are there:

1. application runner.

2. commandline runner.

both runners are functional runners, they have one abstract method.

42. What is @ComponentScan?

ans. it will scan spring bean classes, it will start from basepackage and subpackage.

Basepackage is not but, the class which contains start class.

if @ComponentScan want to scan other packages, we need to override the functionality

@ComponentScan(basePackages={"", ""})

44. Note: Highly recommended to follow proper packaging convention:

ex: companyDomain.projectName.moduleName.layerName.

45. When to @Component and when to @Bean.

ans. If you want IOC should create bean, then go with @Component annotation.

Programmer want to create object explicitly then go with @Bean.

@Component is a Class level annotation, @Bean is method level annotation

46. How to represent java class as a spring bean?

1. @Component

2. @Service

3. @Repository

4. @Controller

5. @RestController

6. @Configuration

7. @Bean

All are the class level annotation, except bean annotation

@Component : it is a general purpose annotation

@Service : stereotype for service layer

@Repository stereotype for persistence layer

@Controller: I web application to represent java class as controller we will use @controller, in C2B communication.

@RestController: I distributed to represent java class as controller we will use @RestController, in B2B communication.

@Configuration: to perform any configuration we will use @configuration. ex: swagger config, Db config,

restTemplate

@Bean: if programmer want to create bean explicitly to use @Bean

Note: it is highly recommended to write @Bean in @configuration.

47. Which annotation will be used for autowiring

ans. @Autowiring and it should be three places, setter method, constructor and variable.

48. If interface have more than 2-implements then loc will get confused to perform DI(ambiguity)

ans. To resolve ambiguity problem we will use @Qualifier

49. @Qualifier, It will use byType mode to inject dependent object

50. @Primary, It will use to resolve ambiguity

Note : If interface have more than one implementation, to resolve ambiguity, if bean have name go for @Qualifier otherwise use @Primary

51. Note: IF we have only one param constructor it optional to write @Autowired annotation. If we have both 0-param & param constructor

we should write @Autowired annotation

52. Among all injection which is recommended.

ans. Field is against OOPS principle, we should not use but real time we will use field only because one line

Recommended to use Constructor injection, Because first dependent bean created then target bean create.

53. SI	vS	CI	vS	FI
1. It is mandatory to specify @Autowired annotation in setter injection. Otherwise DI will not happen (partial injection) Will get null pointer exception		1. It is optional to specify @Autowired if we have only one param constructor.		1. Field
2. Target bean will be created first		2. Dependent bean will be created first then target will create.		

54. Note: @Autowired supports Reference dataTypes only

55. Bean life Cycle:

ans. we have 2-approaches for bean life cycle

1. By implementing interfaces. implements InitializingBean, DisposableBean

2. Annotations @PostConstruct, @PreDestroy

Note: If you use interface implementation for bean life cycle, classes are tightly coupled.