1. Which of the following is not a valid Spring bean scope?

Answers

1. singleton
2. prototype
3. session
4. global

2. In the Spring Bean life cycle, which method is called after dependency injection is done?

Answers

1. destroy()
2. afterPropertiesSet()
3. init()
4. setBeanName()

3. Which interface should a bean implement to get notified of its initialization?

Answers

1. InitializingBean
2. BeanFactoryAware
3. BeanPostProcessor
4. ApplicationContextAware

4. What is the main difference between auto-wiring and explicit wiring in Spring?

Answers

1. Auto-wiring is used only for XML
2. Explicit wiring requires annotations
3. Auto-wiring lets Spring resolve dependencies automatically
4. Explicit wiring is available only in Spring Boot

5. Field injection in Spring is done using:

Answers

1. @Bean
2. @Value
3. @Autowired
4. @ComponentScan

6. In XML configuration, how do you specify auto-wiring by type?

Answers

1. <bean autowire="name">
2. <bean autowire="byType">
3. <bean autowire="byConstructor">
4. <bean autowire="auto">

7. Which annotation is used to disambiguate auto-wiring when multiple beans of the same type are present?

Answers

1. @Autowired
2. @Component
3. @Qualifier
4. @Inject

8. What happens when a Spring bean is defined with prototype scope?

Answers

1. A single instance is created and reused for all requests

2. A new instance is created every time the bean is requested from the container

3. The bean is shared across all sessions

4. Bean is automatically garbage collected after initialization

9. What is the correct order of the Spring Bean life cycle (using annotations and lifecycle interfaces)?

Answers

1. Constructor → @PostConstruct → afterPropertiesSet() → destroy()

2. Constructor → afterPropertiesSet() → @PostConstruct → destroy()

3. Constructor → @PostConstruct → afterPropertiesSet() → @PreDestroy

4. Constructor → @PreDestroy → afterPropertiesSet() → @PostConstruct

10. Which component is used to customize bean creation before and after initialization?

Answers

1. BeanFactory

2. ApplicationContext

3. BeanPostProcessor

4. InitializingBean

11. Which of the following is true about using @Autowired on a field vs on a constructor?

Answers

1. Field injection allows easier unit testing

2. Constructor injection is safer and ensures immutability

3. Field injection is more explicit and preferred

4. Constructor injection cannot be used in Spring Boot

12. What is the default scope of the beans?

Answers

1. Prototype

2. session

3. Request

4. Singleton

13. If multiple beans of the same type exist, and @Autowired is used without @Qualifier, what happens?

Answers

1. Spring throws a NoUniqueBeanDefinitionException

2. Spring randomly picks one of the beans

3. Spring fails to inject and application context does not load

4. Spring injects the last defined bean

14. In Spring, the @Scope("request") scope is available only when:

Answers

1. Running in a standalone Java application

2. Using a servlet-based web application context

3. Declaring the bean with @Component

4. Using Spring Boot

15. What is the purpose of @Primary in Spring Bean configuration?

Answers

1. To delay bean initialization

2. To avoid circular dependency

3. To mark one bean as the default when multiple candidates are available

4. To enforce eager initialization

16. What is the main purpose of Spring Boot?

Answers

1. To replace Spring Framework

2. To simplify Spring application development with auto-configuration and embedded servers

3. To generate Java code from UML diagrams

4. To perform only REST API testing

17. What is the role of SpringApplication.run() in a Spring Boot Java SE app?

Answers

1. It executes the main business logic

2. It starts the Spring context and auto-configuration

3. It compiles the application

4. It creates the WAR file

18. Which of the following are stereotype annotations in Spring?

Answers

1. @Entity, @Table

2. @Autowired, @Qualifier

3. @Component, @Service, @Controller, @Repository

4. @Bean, @Scope

---

19. Which annotation is used to specify the primary key in a JPA entity?

Answers

1. @Primary

2. @Key

3. @Id

4. @GeneratedValue

---

20. What is the purpose of Spring Data JPA?

Answers

1. To write SQL queries manually

2. To simplify data access using repository interfaces and reduce boilerplate code

3. To create embedded databases only

4. To configure application security

21. Which of the following best describes the architecture layers in a Spring Data application?

Answers

1. Controller → Repository → Service

2. Repository → Service → Controller

3. Controller → Service → Repository

4. Service → Repository → Controller

22. What is the role of the @Transactional annotation in Spring?

Answers

1. To enable auto-configuration of Spring Boot

2. To define repository interfaces

3. To manage database transactions programmatically

4. To declare that a method should execute within a transactional context

23. Where should the @Transactional annotation typically be placed?

Answers

1. In the repository interface

2. On the controller class

3. On the service layer methods

4. On the entity class

24. What is no mode of autowiring?

Answers

1. Autowiring by property name.

2. Similar to byType, but type applies to constructor arguments.

3. Default setting which means no autowiring and you should use explicit bean reference for wiring.

4. Spring first tries to wire using autowire by constructor, if it does not work,Spring tries to autowire by byType.

25. Which is not DI?

Answers

1. Setter Based DI

2. Auto-wiring DI

3. Both of the above

4. None of the above.

26. Which layer in Spring handles business logic and typically manages transactions?

Answers

1. Controller Layer

2. Repository Layer

3. Service Layer

4. Entity Layer

27. Which is not application context?

Answers

1. ClassPathXmlApplicationContext

2. AnnotationApplicationContext

3. WebApplicationContext

4. WebPathXmlApplicationContext

28. Which is Stereo Type Annotations?

Answers

1. @Autowired

2. @Controller

3. @PreDestroy

4. @Qualifier

29. What is Spring MVC framework?

Answers

1. Spring MVC framework is Model-Value-Class architecture and used to bind model data with values.

2. The Spring web MVC framework provides model-view-controller architecture and ready components that can be used to develop flexible and loosely c

3. Spring MVC framework is used for Transaction management for Web Applications.

4. Spring MVC framework is used for AOP for Web Applications.

30. How to handle shut down of IoC container?

Answers

1. Using shutdownHook()

2. Using shutdownHandler()

3. Using registerHook()

4. Using registerShutdownHook()

31. What is @Controller annotation?

Answers

1. The @Controller annotation indicates that a particular class serves the role of a controller.

2. The @Controller annotation indicates how to control the transaction management.

3. The @Controller annotation indicates how to control the dependency injection.

4. The @Controller annotation indicates how to control the aspect programming.

---

32. Which of the following is correct about messaging in RESTful web services?

Answers

1. A client sends a message in form of a HTTP Request and server responds in form of a HTTP Response.

2. These messages contain message data and metadata i.e. information about message itself.

3. Both of the above

4. None of the above.

---

33. Spring provides a factory that can export beans annotated with:-

Answers

1. javax.jws.WebService

2. javax.jws.WebServiceProvider

3. all of the mentioned

4. none of the mentioned

---

34. Factory to use our Spring bean as the implementation:-

Answers

1. jaxws:end

2. jaxws:endpoint

3. all of the mentioned

4. none of the mentioned

35. What is the correct execution flow of a Spring REST request?

Answers

1. DispatcherServlet → Service → Controller → Repository

2. Controller → DispatcherServlet → Service → Repository

3. DispatcherServlet → Controller → Service → Repository

4. Repository → Controller → DispatcherServlet → Service

36. What is content negotiation in Spring REST?

Answers

1. Deciding the HTTP method to use

2. Choosing between PUT and POST automatically

3. Determining the best response format (JSON/XML) based on request headers

4. Negotiating with database for data retrieval

37. What is the most common way for a React frontend to consume a Spring REST service?

Answers

1. Using direct JDBC calls from React

2. Fetching data via fetch() or axios from React using RESTful endpoints

3. Importing Java classes into React

4. Using JSP to call backend methods

38. What is bean in Spring?

Answers

1. A component

2. An Object

3. A class

4. A container

39. Spring's core transaction management abstraction is based on the interface _____

Answers

1. TransactionManager

2. PlatformTransactionManager

3. PlatformManager

4. PlatformTransaction

40. In a spring application class , which of the following is not a part of @SpringBootApplication ?

Answers

1. @Configuration

2. @EnableAutoConfiguration

3. @EnableWebMvc

4. @ComponentScan