Welcome to Cert007 - Your Ultimate IT Certification Partner

- Real Exam Questions
- Free Updates
- Expert Support
- Instant Access
- Money-Back Guarantee

Visit us at **https://www.cert007.com/** for more information

Exam : **2V0-72.22**

Title : Spring Professional Develop

Version : DEMO

1.If a class is annotated with @Component, what should be done to have Spring automatically detect the annotated class and load it as a bean? (Choose the best answer.)
A. Ensure a valid bean name in the @Component annotation is specified.
B. Ensure a valid @ComponentScan annotation in the Java configuration is specified.
C. Ensure a valid @Scope for the class is specified.
D. Ensure a valid @Bean for the class is specified.
Answer: B
Explanation:
The @Component annotation indicates that a class is a candidate for auto-detection by Spring and can be registered as a bean in the application context. However, to enable this feature, the Java configuration class must also have the @ComponentScan annotation, which tells Spring where to look for annotated components.
Reference: https://docs.spring.io/spring-framework/docs/current/reference/html/core.html#beans-stereotype-annotations

2.Which two options will inject the value of the daily.limit system property? (Choose two.)
A. @Value("#{daily.limit}")
B. @Value("$(systemProperties.daily.limit)")
C. @Value("$(daily.limit)")
D. @Value("#{systemProperties['daily.limit']}")
E. @Value("#{systemProperties.daily.limit}")
Answer: CD
Explanation:
The @Value annotation can be used to inject values from external sources into fields, constructor parameters, or method parameters. To inject a system property, the annotation can use either the ${…} placeholder syntax or the #{…} SpEL expression syntax. The former is simpler and more concise, while the latter is more powerful and flexible. Both syntaxes can access the systemProperties map, which contains all the system properties as key-value pairs.
Reference:
https://docs.spring.io/spring-framework/docs/current/reference/html/core.html#beans-value-annotations:
https://docs.spring.io/spring-framework/docs/current/reference/html/core.html#expressions:
https://docs.spring.io/spring-framework/docs/current/reference/html/core.html#expressions-beandef-xml-based

3.Which two options are REST principles? (Choose two.)
A. RESTful applications use a stateless architecture.
B. RESTful application use HTTP headers and status codes as a contract with the clients.
C. RESTful applications cannot use caching.
D. RESTful application servers keep track of the client state.
E. RESTful applications favor tight coupling between the clients and the servers.
Answer: AB
Explanation:
REST stands for Representational State Transfer, which is an architectural style for designing web services that adhere to certain principles. One of these principles is statelessness, which means that

each request from a client to a server must contain all the information necessary to understand the request, and that no session state is maintained on the server side. Another principle is uniform interface, which means that clients and servers communicate using a standardized protocol, such as HTTP, and use its features, such as headers and status codes, to exchange information about the resources and their representations.
Reference:
https://www.ibm.com/cloud/learn/rest:
https://restfulapi.net/statelessness/:
https://restfulapi.net/uniform-interface/

4.Which option is true about use of mocks in a Spring Boot web slice test? (Choose the best answer.)
A. Mocking a Spring Bean requires annotating it with @MockBean annotation.
B. If a Spring Bean already exists in the web slice test spring context, it cannot be mocked.
C. Mocks cannot be used in a Spring Boot web slice test.
D. Mocking a Spring Bean requires annotating it with @Mock annotation.
Answer: A
Explanation:
A web slice test is a type of test in Spring Boot that focuses on testing only the web layer of an application, without starting a full server or requiring a real database. To isolate the web layer from other dependencies, such as services or repositories, mocks can be used to simulate their behavior and return predefined responses. To create and inject a mock for an existing bean in the application context, the @MockBean annotation can be used on a field or a parameter in the test class. This annotation will replace any existing bean of the same type with a mock created by Mockito.
Reference: https://docs.spring.io/spring-boot/docs/current/reference/html/features.html#features.testing.spring-boot-applications.testing-autoconfigured-web-test: https://docs.spring.io/spring-boot/docs/current/api/org/springframework/boot/test/mock/mockito/MockBean.html:
https://site.mockito.org/
Reference: https://tanzu.vmware.com/developer/guides/spring-boot-testing/

5.Which two statements are true regarding Spring Security? (Choose two.)
A. Access control can be configured at the method level.
B. A special Java Authentication and Authorization Service (JAAS) policy file needs to be configured.
C. Authentication data can be accessed using a variety of different mechanisms, including databases and LDAP.
D. In the authorization configuration, the usage of permitAll () allows bypassing Spring security completely.
E. It provides a strict implementation of the Java EE Security specification.
Answer: AC
Explanation:
Spring Security is a framework that provides comprehensive security services for Java applications, such as authentication, authorization, encryption, session management, and more. One of its features is method security, which allows applying access control rules at the method level using annotations or XML configuration. Another feature is authentication, which is the process of verifying the identity of a

user or a system. Spring Security supports various authentication mechanisms, such as username and password, tokens, certificates, etc., and can access authentication data from different sources, such as databases, LDAP directories, in-memory stores, etc.
Reference:
https://docs.spring.io/spring-security/site/docs/current/reference/html5/:
https://docs.spring.io/spring-security/site/docs/current/reference/html5/#jc-method:
https://docs.spring.io/spring-security/site/docs/current/reference/html5/#servlet-authentication
Reference: https://www.baeldung.com/security-none-filters-none-access-permitAll