Spring Certified #1

A question lead guide to prepare Spring certification

Configuration

Which annotation is used to indicate that a class is a Spring configuration class in a Spring Boot application?

- ➔ **@TestContext**
- ➔ **@TestConfiguration**
- ➔ **@ConfigurationProperties**
- ➔ **@Configuration**

@Configuration

The `@Configuration` annotation is used to indicate that a class is a Spring configuration class in a Spring Boot application. This annotation is used to declare one or more `@Bean` methods that are used to produce Spring beans to be managed by the Spring container. The `@Configuration` annotation can be used in any class that needs to define Spring beans, including classes in the src/test/java directory that are used for testing.

https://docs.spring.io/spring-framework/docs/5.3.6/javadoc-api/org/springframework/context/annotation/Configuration.html

Spring AOP

What is the mechanism used to implement Declarative Transaction Management in Spring Applications?

- ➔ **Spring AOP**
- ➔ **Spring Security**
- ➔ **Spring Data**
- ➔ **Spring MVC**

Spring AOP

The correct answer is **Spring AOP** (Aspect-Oriented Programming).

Spring's Declarative Transaction Management is implemented using AOP, which provides a way to modularize cross-cutting concerns, such as transaction management, into reusable aspects.

By using annotations like `@Transactional`, developers can declaratively specify which methods should be wrapped in a transaction. Spring AOP takes care of the rest, automatically handling transaction boundaries and rollbacks.

Spring Security is used for authentication and authorization, Spring Data is used for accessing databases, and Spring MVC provides a model-view-controller architecture for building web applications that separate the application's concerns.

https://docs.spring.io/spring-framework/docs/5.3.6/javadoc-api/org/springframework/transaction/annotation/Transactional.html

What are the advantages of using @MockBean instead of @Mock in Spring Boot tests?

- ➔ **@MockBean allows mocking of any object, even if it is not a Spring bean.**
- ➔ **@MockBean provides context-sensitive mocking for more flexible testing.**
- ➔ **@MockBean improves the performance of test execution.**
- ➔ **@MockBean is a more lightweight alternative to @Mock.**

@MockBean improves the performance of test execution.

`@MockBean` simplifies the creation of mock objects in tests by automatically creating and adding the mock object to the Spring context. This reduces boilerplate code and makes tests easier to read and write.

`@MockBean` provides context-sensitive mocking, which means that the mock object will be created with the appropriate configuration for the current testing context. This allows for more flexible and targeted testing of specific components, without affecting the behavior of other components.

`@MockBean improves the performance of test execution.` This is false. Using @MockBean or @Mock does not affect the performance of test execution in any significant way.

`@MockBean is a more lightweight alternative to @Mock.`This is false. @MockBean is not a more lightweight alternative to @Mock, but rather a specific implementation designed for Spring Boot tests.

`@MockBean allows mocking of any object, even if it is not a Spring bean.` This is false. @MockBean is specifically designed for mocking Spring beans and cannot be used to mock any object.

https://docs.spring.io/spring-boot/docs/2.5.2/api/org/springframework/boot/test/mock/mockito/MockBean.html

https://site.mockito.org/javadoc/current/org/mockito/Mock.html bitly

https://bit.ly/2v7222