Contents

[AOP 1](#_Toc13348182)

[Joint point: a point in the execution of a program such as a method call or exception thrown. 1](#_Toc13348183)

[It is described as a stage during the execution of a program, such as the execution of a method or the handling of an exception .In spring AOP, it always represents a method execution. 1](#_Toc13348184)

[Pointcut: An expression that selects one or more join points 1](#_Toc13348185)

[Advice: code to be executed at each selected join point 1](#_Toc13348186)

[Aspect: A module that encapsulates pointcuts and advice 1](#_Toc13348187)

[Weaving: Technique by which aspects are combined with main code 1](#_Toc13348188)

[ProceedingJoinPoint 3](#_Toc13348189)

[Code Example 3](#_Toc13348190)

[Special Note 3](#_Toc13348191)

[Spring proxy feature: 4](#_Toc13348192)

[Testing 4](#_Toc13348193)

[By default, once loaded configured ApplicationContext can be reused for each testing. 5](#_Toc13348194)

[Which of these are valid ways of creating a shared application context, when testing with JUnit?  Select all that apply. 6](#_Toc13348195)

[Container 6](#_Toc13348196)

[InitializingBean interface 6](#_Toc13348197)

[Database and Transactions 6](#_Toc13348198)

[In which of these database system isolation levels can non-repeatable reads occur? Select all that apply? 6](#_Toc13348199)

# AOP

## Joint point: a point in the execution of a program such as a method call or exception thrown.

## It is described as a stage during the execution of a program, such as the execution of a method or the handling of an exception .In spring AOP, it always represents a method execution.

## Pointcut: An expression that selects one or more join points

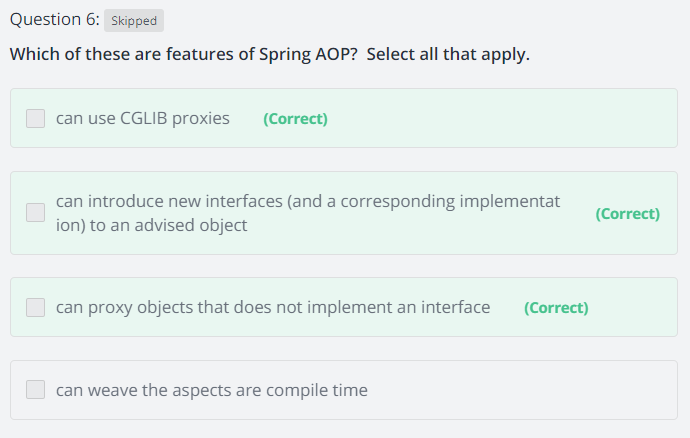
## Advice: code to be executed at each selected join point

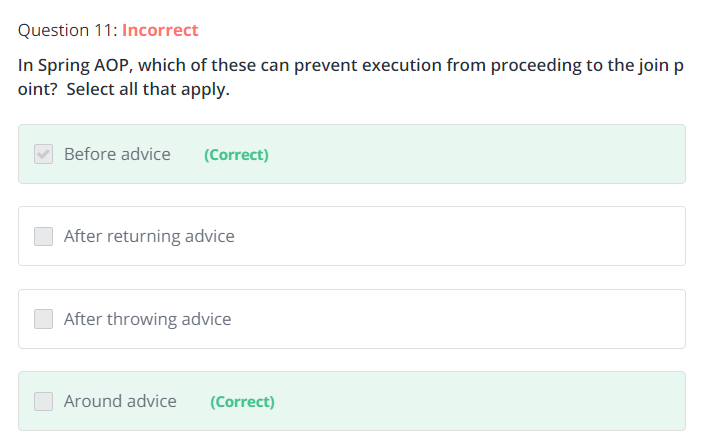
## Aspect: A module that encapsulates pointcuts and advice

It is described as a modularization of a concern that cut across multiple classes or across an entire application,

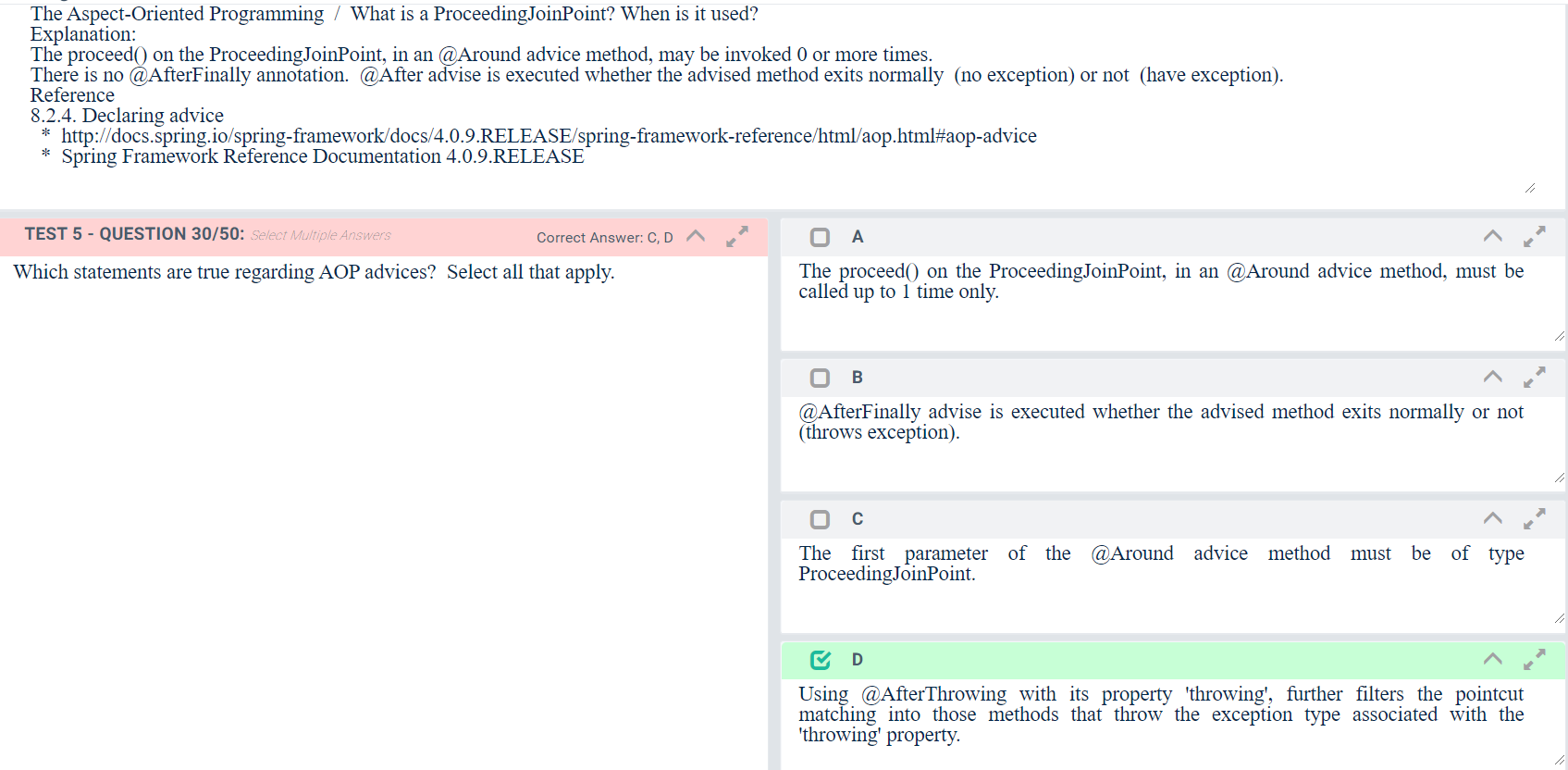
## Weaving: Technique by which aspects are combined with main code

**Spring AOP performs weaving at runtime**  
  
**References**  
[8.1.1. AOP concepts  
     http://docs.spring.io/spring-framework/docs/4.0.9.RELEASE/spring-framework-reference/html/aop.html#aop-introduction-defn](http://docs.spring.io/spring-framework/docs/4.0.9.RELEASE/spring-framework-reference/html/aop.html#_blank)

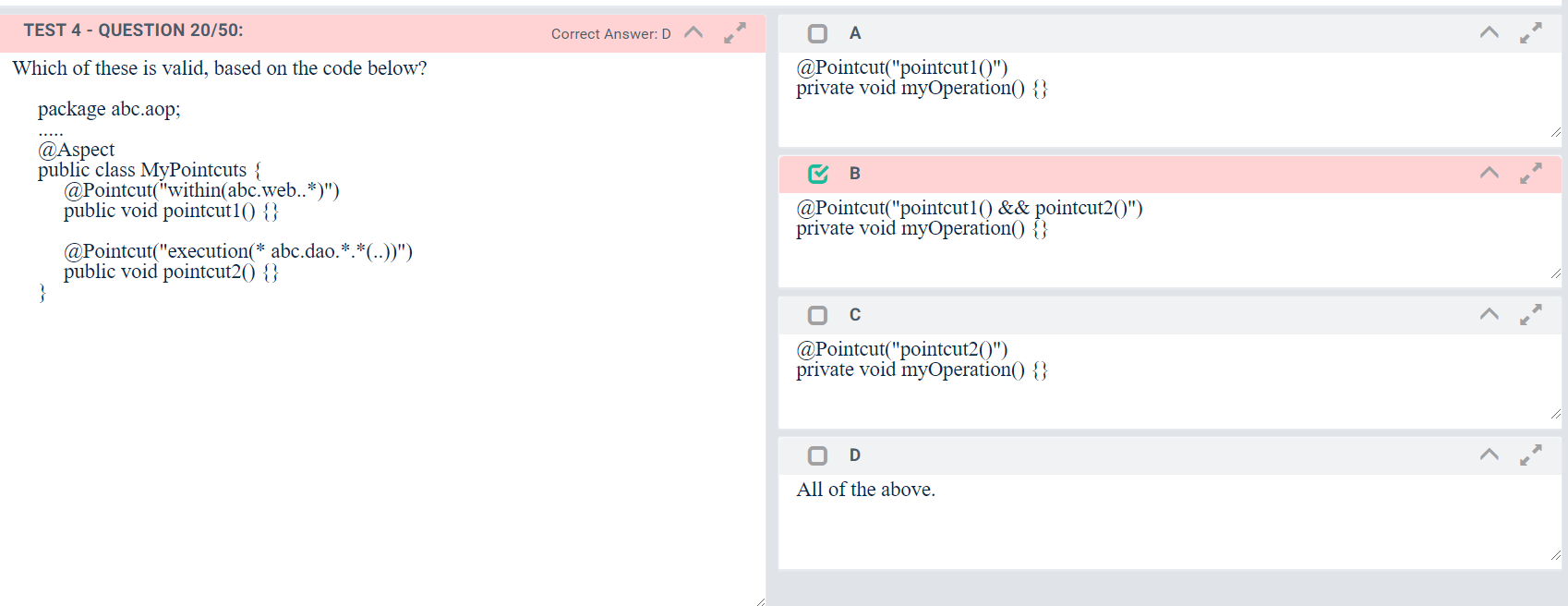




## ProceedingJoinPoint



## Code Example

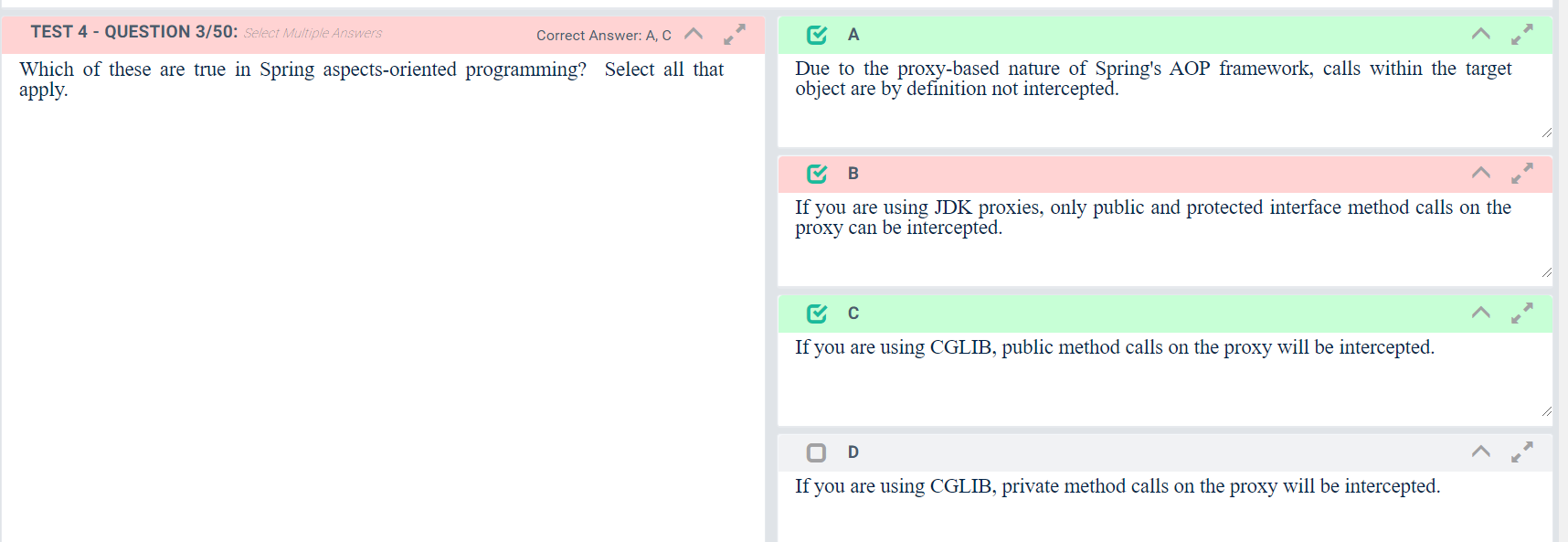


## Special Note

Due to the proxy-based nature of Spring’s AOP framework, calls within the target object are by definition notintercepted. For JDK proxies, only public interface method calls on the proxy can be intercepted. With CGLIB, public and protected method calls on the proxy will be intercepted, and even package-visible methods if necessary. However, common interactions through proxies should always be designed through public signatures.

Note that pointcut definitions are generally matched against any intercepted method. If a pointcut is strictly meant to be public-only, even in a CGLIB proxy scenario with potential non-public interactions through proxies, it needs to be defined accordingly.

If your interception needs include method calls or even constructors within the target class, consider the use of Spring-driven [native AspectJ weaving](https://docs.spring.io/spring/docs/5.0.5.RELEASE/spring-framework-reference/core.html#aop-aj-ltw) instead of Spring’s proxy-based AOP framework. This constitutes a different mode of AOP usage with different characteristics, so be sure to make yourself familiar with weaving first before making a decision.

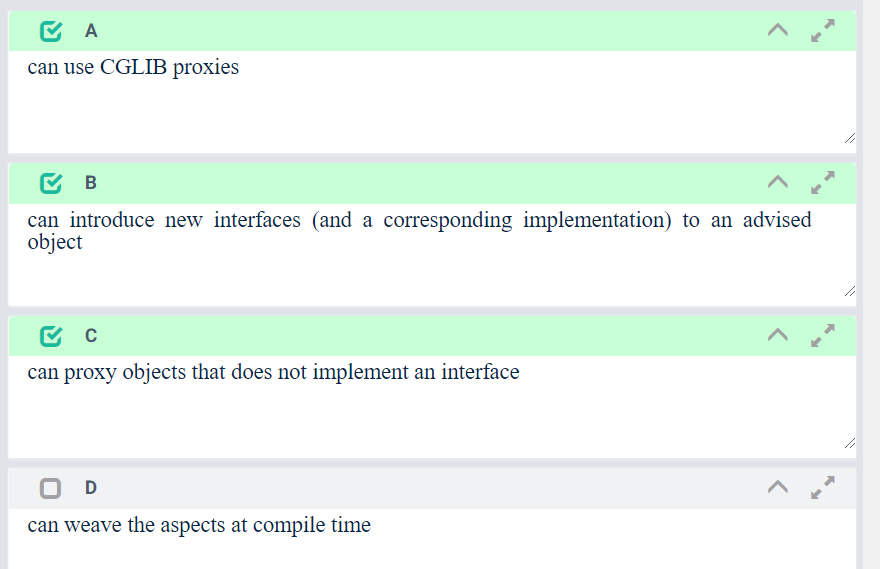


## Spring proxy feature:

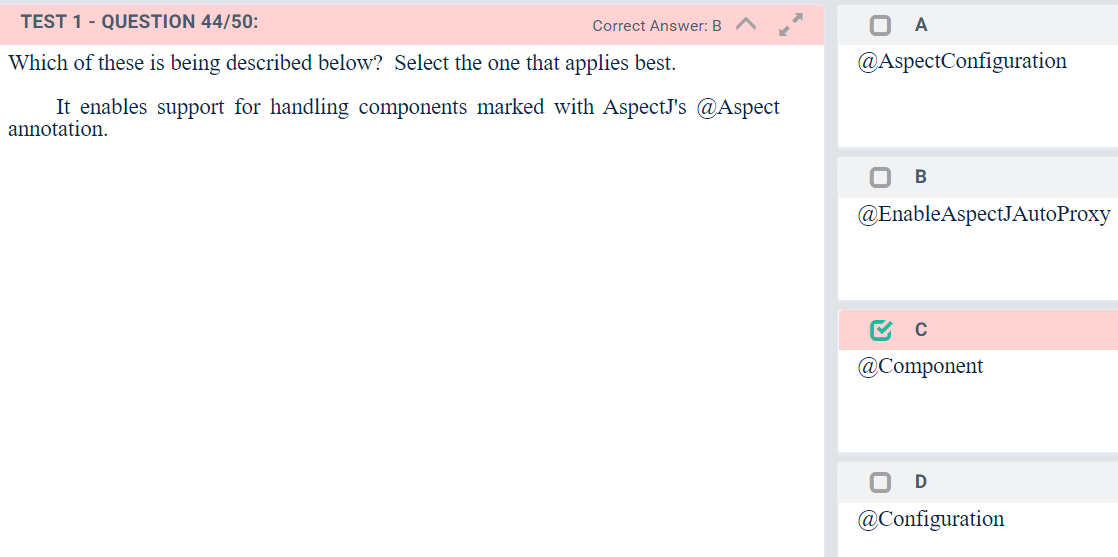
If a proxied method is called directly by another method from inside the same class, the proxy of that proxied

method will not be triggered.

## Which of these are features of Spring AOP? Select all that apply?

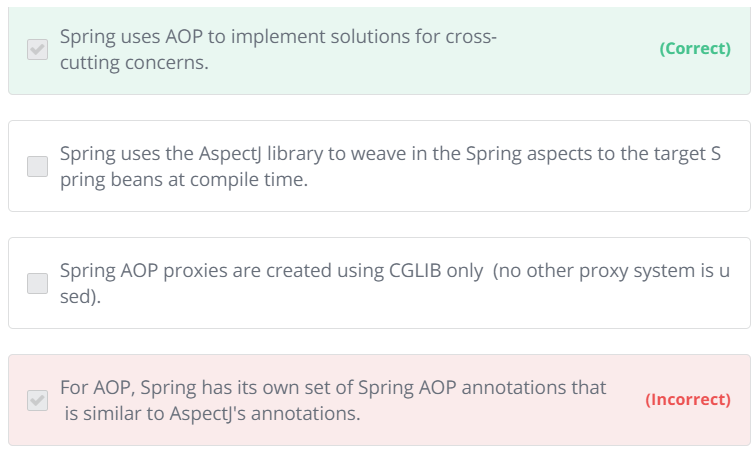


## @EnableAspectJAutoProxy



## Spring does not have separate Spring AOP Annotations.

## Which of these are correct with regards to Spring AOP?  Select all that apply.

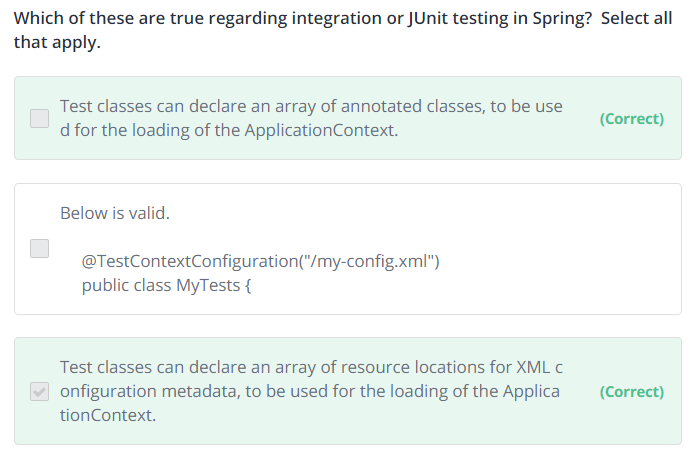


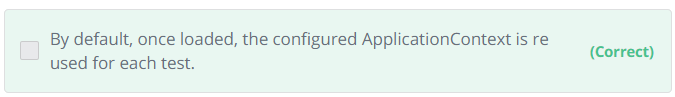
### Explanation

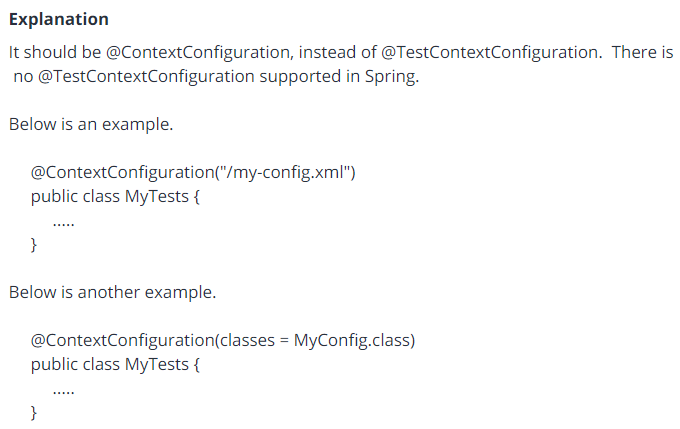
### Spring uses the AspectJ library, but not for weaving in Spring aspects into target Spring beans.  Spring AOP proxies are created using CGLIB proxies or JDK dynamic proxies. For AOP, Spring reuses AspectJ's annotations. References [8.1. Introduction      http://docs.spring.io/spring-framework/docs/4.0.9.RELEASE/spring-framework-reference/html/aop.html#aop-introduction](http://docs.spring.io/spring-framework/docs/4.0.9.RELEASE/spring-framework-reference/html/aop.html#_blank)

# Testing

## By default, once loaded configured ApplicationContext can be reused for each testing.







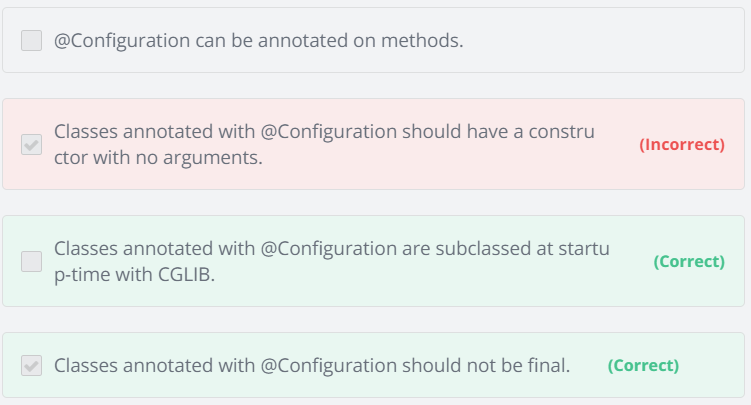
## Which of these are valid ways of creating a shared application context, when testing with JUnit?  Select all that apply.

<https://www.udemy.com/pivotal-certified-spring-professional-43-practice-tests/learn/quiz/4362250/result/184868670#overview>

# Container

## @Configuration can be annotated on classes only.

## The Spring Reference says:      "This is where the magic comes in: All @Configuration classes are subclassed at startup-time with CGLIB."



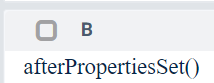
## Classpath application context prefix ("classpath\*:conf/myContext.xml")

## special prefix that specifies that all classpath resources that match the given name must be obtained?

Below is an example.  
  
     ApplicationContext ctx = new ClassPathXmlApplicationContext("classpath\*:conf/myContext.xml");  
  
**References**  
[The Classpath\*: portability classpath\*: prefix  
     http://docs.spring.io/spring-framework/docs/4.0.9.RELEASE/spring-framework-reference/html/resources.html#resources-classpath-wildcards](http://docs.spring.io/spring-framework/docs/4.0.9.RELEASE/spring-framework-reference/html/resources.html#_blank)

## InitializingBean interface





# Spring Beans Scopes

## Prototype scopes:

### For prototype-scoped beans, the client is responsible for cleaning up after the usage of the prototype bean. Spring doesnt keep further record of the mentioned bean after handing it to the client, unless bean post processors are added.

### By default, when a singleton bean depends on a prototype bean, there will be an object instance of the prototype bean, and it is that same object instance that will be supplied to all users of the concerned singleton bean.

### Which ones are true about scopes? Select all that apply.

**Which ones are true about scopes?  Select all that apply.**

* ​

For prototype-scoped beans, the client is responsible for cleaning up after the usage of the prototype bean.  Spring doesnt keep further record of the mentioned bean after handing it to the client, unless bean post processors are added.

**(Correct)**

* ​

By default, when a singleton bean depends on a prototype bean, there will be an object instance of the prototype bean, and it is that same object instance that will be supplied to all users of the concerned singleton bean.

**(Correct)**

## Parent bean and child bean

## 

#### Explanation

If the bean is marked 'abstract', the 'class' is optional.  
  
If the bean is marked 'abstract',  it cannot be instantiated on its own.  
  
When the ApplicationContext pre-instantiates singletons, the ApplicationContext ignores the beans that has the attribute 'abstract="true"'.  
  
The Spring Reference says:  
     "The remaining settings are always taken from the child definition: depends on, autowire mode, dependency check, singleton, lazy init."  
  
**References**  
[4.7. Bean definition inheritance  
     http://docs.spring.io/spring-framework/docs/4.0.9.RELEASE/spring-framework-reference/html/beans.html#beans-child-bean-definitions](http://docs.spring.io/spring-framework/docs/4.0.9.RELEASE/spring-framework-reference/html/beans.html#_blank)

# Database and Transactions

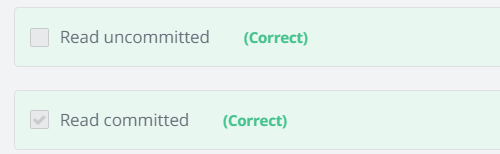
## Repeatable read , phantom read and non-repeatable read?

<https://en.wikipedia.org/wiki/Isolation_%28database_systems%29#Isolation_levels>

<https://en.wikipedia.org/wiki/Isolation_(database_systems)#Phantom_reads>

<https://en.wikipedia.org/wiki/Isolation_(database_systems)#Non-repeatable_reads>

## In which of these database system isolation levels can non-repeatable reads occur? Select all that apply?



#### Explanation

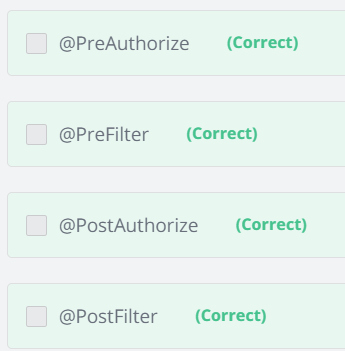
Here is a brief summary of the characteristics of the isolation levels.  
  
     Read uncommitted  
          dirty read can occur:  Yes  
          non-repeatable read can occur:  Yes  
          phantom read can occur:  Yes  
  
     Read committed  
          dirty read can occur:  No  
          non-repeatable read can occur:  Yes  
          phantom read can occur:  Yes  
  
     Repeatable reads  
          dirty read can occur:  No  
          non-repeatable read can occur:  No  
          phantom read can occur:  Yes  
  
     Serializable  
          dirty read can occur:  No  
          non-repeatable read can occur:  No  
          phantom read can occur:  No

**References**  
[Isolation (database systems)  
     http://en.wikipedia.org/wiki/Isolation\_%28database\_systems%29#Isolation\_levels](http://en.wikipedia.org/wiki/Isolation_%28database_systems%29#_blank)  
  
[Enum Isolation  
     http://docs.spring.io/autorepo/docs/spring/4.0.9.RELEASE/javadoc-api/org/springframework/transaction/annotation/Isolation.html](http://docs.spring.io/autorepo/docs/spring/4.0.9.RELEASE/javadoc-api/org/springframework/transaction/annotation/Isolation.html#_blank)

## Which of these can be used so that the @Transactional in the code below will work?      @Transactional      public class MyServiceImpl implements MyService {

# Security

## Which of these Spring Security annotations can use SpEL?  Select all that apply.



#### Explanation

Below is an example.  
  
     @PreAuthorize("hasRole('ROLE\_USER')")  
     public void sendComment(Comment comment);  
  
**References**  
[3.3.1. @Pre and @Post Annotations  
     http://docs.spring.io/spring-security/site/docs/3.2.5.RELEASE/reference/htmlsingle/#el-pre-post-annotations](http://docs.spring.io/spring-security/site/docs/3.2.5.RELEASE/reference/htmlsingle/#_blank)

# MVC

## Scope

### What is the default scope in the web context?

Ans: Singleton

# REST

## Message Converters Table

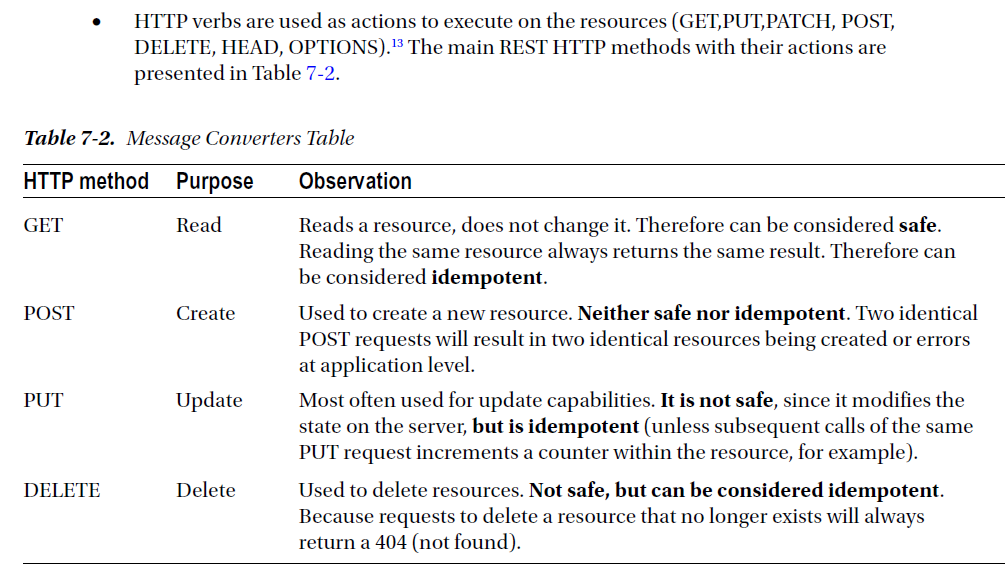
### Things to remember: The REST architectural style describes best practices to expose web services over HTTP; it is used not only as a transport but as an application protocol

#### Web applications are not used only by browser clients. Programmatic clients can connect using HTTP

#### (e.g., mobile applications and basically any application that was developed to be able to request and

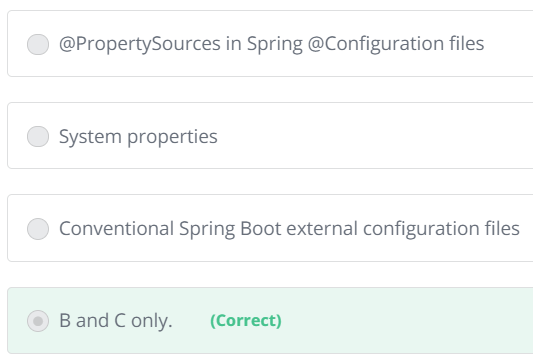
#### receive data using HTTP as communication protocol). The REST architectural style describes best practices to expose web services over HTTP; it is used not only as a transport but as an application protocol. The

#### following HTTP specifications are used:



# Spring Boot

## Control Logging / In Spring Boot, on which areas below, can logging be controlled?



The below happens after the configuration of the logging.  That is why it is not a stage that logging can be controlled.  
  
     @PropertySources in Spring @Configuration files