Spring Professional Practice Questions

Javin Paul @javinpaul

Table Of Contents

Spring Professional Practice Questions

Version 1.0 - August 2022

Overview

Hello guys, preparing for IT certification like Oracle's Java certification or Vmware's Spring Certification required a lot of hard-work. I have seen many experienced Java developers failing these certifications and losing money and time due to over-confidence and lack of preparation.

A structured preparation involves reading books, joining course and doing practice questions. When it comes to Spring certification, practice questions are quite hard to find and that's why I have created this Spring certification Practice Test bank.

In this eBook you will find around 200 questions on different Spring related topics as per Spring Professional Certification and 1 full length test to further check your preparation level. You can use these Practice Test to find your current preparation level as well as your strong and weak areas.

I suggest to work on your weak areas where you have difficulty answering these practice questions so that you can score high marks on real exams.

AOP

1. Which of the following are cross cutting concerns where Spring AOP can be useful?

1. Security

2. File access

3. Parallelization

4. Logging

Correct Answer: 1, 4

Explanation: "There are many valid AOP use cases. Other use cases include: performance monitoring, statistics, transactions, caching.

2. Which of the following operators can be used to combine multiple pointcut expressions?

1. and

2. or

3. &&

4. ||

Correct Answer: 3, 4

Explanation: "You can combine pointcut expressions by using &&, || and !. You can also refer to pointcut expressions by name.

3. _____ is an additional behavior, typically a cross cutting concern, that is to be executed at certain places (at join Points) in a program.

1. Advice

2. Aspect

3. Weaving

4. Pointcut

Correct Answer: 1

Explanation: "Advice: Action taken by an aspect at a particular join point. Different types of advice include around, before, and after advice. (Advice types are discussed later.) Many AOP frameworks, including spring, model an advice as an interceptor and maintain a chain of interceptors around the join point.

4. The methods in which of the following classes are intercepted when using the following pointcut expression "within (com.xyz.example..*)"

1. com.xyz.example.rest.http.MyController

2. com.xyz.example.service.MyService

3. com.xyz.security.SecurityConfig

4. com.xyz.example.Application

Correct Answer: 1, 2, 4

Explanation: "Two consecutive dots (.) Recursively matches all sub-packages.

The pointcut operator within mathces all methods in the specified packages.

5. Which of the following types of AOP advice will execute code after the advised method regardless if it completes normally or throws an exception?

1. @AfterReturning

2. @AfterThrowing

3. @Around

4. @After

Correct Answer: 4

Explanation: "After (finally) advice will execute regardless if an exception was thrown or not.

6. "In Spring AOP, two dots (..) Instead of method arguments, like in the example

below, matches _____.
Example: execution(* transfer(..))"

1. No arguments

2. One or more arguments

3. Multiple arguments

4. Any number of arguments

Correct Answer: 4

Explanation: "Method arguments (..) signify zero or more arguments.

7. Which of the following types of AOP advice can be used to execute code before the advised method?

1. @AfterThrowing

2. @AfterReturning

3. @Before

4. @Around

Correct Answer: 3, 4

Explanation: "All types of After advice execute exclusively after the advised method has executed.

Around advice runs both before and after. Before

advice runs only before the method execution.

8. Which of the following pointcut expressions allow picking method access specifiers?

1. target

2. bean

3. execution

4. within

Correct Answer: 3

Explanation: "Execution allows optionally picking the access specifier, for example execution(public * *(..))

9. _____ complements Spring IoC by focusing on _____

1. Spring AOP, Inversion of Control

2. JPA, automated testing

3. Spring AOP, crosscutting concerns

4. Spring Actuator, Reactive Programming

Correct Answer: 3

Explanation: "Aspects enable the modularization

of concerns (such as transaction management) that cut across multiple types and objects. (Such concerns are often termed “crosscutting” concerns in AOP literature.)

10. Which of the following types of AOP advice may potentially be used to suppress an exception thrown in the advised method?

1. @Before

2. @Around

3. @AfterReturning

4. @AfterThrowing

Correct Answer: 2, 4

Explanation: “Around and after throwing can both suppress exceptions thrown in the advised exception.

After returning advice will only be executed if the advised method has not thrown any exception (thus no exception suppression can happen).

11. Which of the following statements are true about Spring AOP?

1. Calls within the same object can be intercepted.

2. When using CGLIB proxies, public method calls

can be intercepted

3. When using JDK Proxies, only public and protected interface method calls can be intercepted

4. When using CGLIB proxies, private method calls can be intercepted

Correct Answer: 1, 2

Explanation: "Calls within the same objects will not be intercepted because Spring AOP is implemented using Proxies. When using JDK Proxies, only public and protected interface method calls are intercepted.

12. In Spring AOP, _____ can be of multiple types, such as "around", "before" and "after".

1. Advice

2. Join point

3. Aspect

4. Pointcut

Correct Answer: 1

Explanation: Spring AOP includes the following types of advice:

Before advice: Advice that runs before a join point but that does not have the ability to prevent execution flow proceeding to the join point (unless it throws an exception).

After returning advice: Advice to be run after a join point completes normally (for example, if a method returns without throwing an exception).

After throwing advice: Advice to be run if a method exits by throwing an exception.

After (finally) advice: Advice to be run regardless of the means by which a join point exits (normal or exceptional return).

Around advice: Advice that surrounds a join point such as a method invocation. This is the most powerful kind of advice. Around advice can perform custom behavior before and after the method invocation. It is also responsible for choosing whether to proceed to the join point or to shortcut the advised method execution by returning its own return value or throwing an exception.

13. What does the abbreviation AOP stand for?

1. Access Operational Program

2. Aspect Oriented Programming

3. Application Oriented Programming

4. Approved Operating Program

Correct Answer: 2

Explanation: Aspect-oriented Programming (AOP) complements Object-oriented Programming (OOP) by providing another way of thinking about program structure. The key unit of modularity in OOP is the class, whereas in AOP the unit of modularity is the aspect.

14. The methods in which of the following classes are intercepted when using the following pointcut expression "within(com.xyz.example.service.*)"

1. com.xyz.example.rest.http.MyController

2. com.xyz.example.service.MyService

3. com.xyz.service.payment.PaymentService

4. com.xyz.example.Application

Correct Answer: 1,2,4

Explanation: Two consecutive dots (.) recursively matches all subpackages.

The pointcut operator within metches all methods in the specified packages.

15. Which of the following types of AOP advice can be used to prevent execution of

the advised method?

1. @Before

2. @AfterThrowing

3. @After

4. @Around

Correct Answer: 1,4

Explanation: If the Before advice throws an exception, the join point will not be called.

By not calling the 'proceed()' on the ProceedingJoinPoint parameter of the Around advice method, the join point (target method) will not be executed.

'After returning advice' and 'After throwing advice' are involved only after the execution of the join point.

16. What is an aspect in Spring AOP?

1. An expression that selects on or more join point

2. Technique by which aspect are combined with main code.

3. Code to be executed at each selected join poin

4. A module that encapsulate pointcuts and advice

Correct Answer: 4

Explanation: Join point: a point in the execution of a program such as a method call or exception thrown

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

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

Boot Actuator

1. Which of the following are metrics provided by Spring Boot Actuator?

1. HTTP Request response time

2. Memory usage

3. Garbage collection statistics

4. Number of user sessions

Correct Answer: 1,2,3

Explanation: "Some examples of metrics that can be found in a Spring Boot application are:

1. Response time of HTTP requests.

2. Number of active connections in a database connection pool.

3. Memory usage.

4. This can be for instance heap memory usage.

5. Garbage collection statistics.

Learn more: - _

2. Which of the following are health indicators provided by Spring Boot Actuator by default?

1. ConnectionHealthIndicator

2. DiskSpaceHealthIndicator

3. ApplicationHealthIndicator

4. ElasticsearchHealthIndicator

Correct Answer: 2,4

Explanation: "DiskSpaceHealthIndicator - Verifies that available disk space is above a certain threshold.

ElasticsearchHealthIndicator - Verifies the availability of an Elasticsearch cluster.

ConnectionHealthIndicator and ApplicationHealthIndicator do not exist.

3. Which of the following are endpoints provided by Spring Boot Actuator?

1. management

2. beans

3. graph

4. info

Correct Answer: 2,4

Explanation:

“beans: Lists the Spring beans in the application.

info: Arbitrary application information.

Spring Boot Actuator, by itself, does not provide metrics visualization.”

4. Which of the following are endpoints provided by Spring Boot Actuator?

1. mappings

2. users

3. flyway

4. liquibase

Correct Answer: 1,3,4

Explanation: “mappings: Lists the @ RequestMapping paths of the application.

flyway: Lists applied Flyway database migrations.

liquibase: Lists applied Liquibase database migrations.

Spring Boot Actuator does not have an endpoint called users.”

5. The property _____ can be used to expose all available actuator endpoints?

1. management.endpoints.enable=true

2. management.endpoints.enabled-by-default=true

3. management.endpoints.web.include=*

4. management.endpoints.web.exposure.include=*

Correct Answer: 2, 4

Explanation: "The configuration property enabled-by-default specifies whether Actuator endpoints should be exposed by default. The property web.exposure.include specifies a list of exposed endpoints (* is a wildcard which selects all endpoints)

6. Which of the following endpoints are exposed over HTTP by default?

1. shutdown

2. info

3. health

4. beans

Correct Answer: 2, 3

Explanation: By default, only the endpoints health and info are exposed via HTTP.

7. How can the Health Indicator status severity order be changed?

1. Using the @Ordered annotation

2. Using the @Primary annotation

3. Using the property management.health.status.order

4. It cannot be changed.

Correct Answer: 3

Explanation: "The @Ordered and @Primary annotations are used to change the order in which Spring beans are instantiated.

8. _____ is a Health Indicator Status provided in Spring Boot Actuator?

1. UP

2. UNRESPONSIVE

3. DOWN

4. UNKNOWN

Correct Answer: 1, 3, 4

Explanation: "UP, DOWN and UNKNOWN are Health Indicator Statuses provided and used by Spring Boot Actuator by default.

9. What is measuring different aspects of the application for determining the performance of the application at different points in time, under different conditions?

1. Metrics

2. Management

3. Auditing

4. Monitoring

Correct Answer: 1

Explanation: "A software metric is a standard of measure of a degree to which a software system or process possesses some property. Even if a metric is not a measurement (metrics are functions, while measurements are the numbers obtained by the application of metrics), often the two terms are used as synonyms.

10. Which of the following are reasons to use a 3rd party external monitoring system?

1. Monitoring multiple applications at once

2. Visualization of monitoring data

3. Monitoring of JVM metrics

4. Exposing endpoints for monitoring over a network

Correct Answer: 1, 2

Explanation: "Monitoring of JVM metrics is already provided by Spring Boot Actuator. Endpoints for monitoring are already exposed over the network using the JMX or HTTP protocol by Spring Boot Actuator.

Limitations of Spring Boot Actuator include monitoring multiple applications (since each runs its own instance of Actuator). Other reasons for using 3rd party external monitoring systems include:

Retain monitoring data over time.

This gives several subsequent opportunities, some of which are listed below.

Allow for querying monitoring data.

Allow for visualization of monitoring data.

Enable alerting based on monitoring data.

Allow for analysis of monitoring data to find trends and to discover anomalies.

11. _____ is an endpoint exposed by Spring Boot Actuator by default?

1. beans

2. info

3. metrics

4. health

Correct Answer: 2, 4

Explanation: "By default, Spring Boot Actuator exposes the endpoints health and info.

12. Which of the following can be used to monitor a Spring Boot application?

1. Spring Boot Devtools

2. Spring Boot Initializr

3. Spring Boot Starter

4. Spring Boot Actuator

Correct Answer: 4

Explanation: One of the purposes of Actuator is to monitor Spring Boot Applications.

13. How can auditing be enabled for Spring

Boot Actuator?

1. The @EnableAuditing annotation

2. Setting the property actuator.audit.
enabled=true

3. Declaring a bean of type AuditEventRepository

4. None of the above.

Correct Answer: 3

Explanation: “Once Spring Security is in play, Spring Boot Actuator has a flexible audit framework that publishes events (by default, “authentication success”, “failure” and “access denied” exceptions). This feature can be very useful for reporting and for implementing a lock-out policy based on authentication failures.

Auditing can be enabled by providing a bean of type AuditEventRepository in your applicationâ€™s configuration. For convenience, Spring Boot offers an InMemoryAuditEventRepository. InMemoryAuditEventRepository has limited capabilities and we recommend using it only for development environments. For production environments, consider creating your own alternative AuditEventRepository implementation.

The @EnableAuditing does not exist.

Configuration Properties relating to actuator are

all under the key ""management"". For example: management.auditevents.enabled = true (default).

14. Which of the following are endpoints provided by Spring Boot Actuator?

1. shutdown

2. env

3. stacktrace

4. loggers

Correct Answer: 1, 2, 4

Explanation:

"shutdown: Shutdown of the application.

env: Lists properties from the Spring ConfigurableEnvironment.

loggers: Allows for viewing and modification of the application's log configuration.

The endpoint stacktrace does not exist."

15. Which of the following endpoints are exposed over JMX by default?

1. shutdown

2. beans

3. health

4. info

Correct Answer: 1,2,3,4

Explanation: All Actuator endpoints are exposed on JMX by default.

16. What does HTTP Tracing mean in the context of Spring Boot Actuator?

1. Logging all actions initiated via HTTP request using the TRACE level

2. Logging http request and requester IP address

3. Logging failed HTTP requests

4. Logging HTTP requests and responses.

Correct Answer: 4

Explanation: "HTTP Tracing can be enabled by providing a bean of type HttpTraceRepository in your application's configuration. For production environments, use of a production-ready tracing or observability solution, such as Zipkin or Spring Cloud Sleuth, is recommended. Alternatively, create your own HttpTraceRepository that meets your needs.

The httptrace endpoint can be used to

obtain information about the request-response exchanges that are stored in the HttpTraceRepository.

Boot AutoConfig

1. Spring MVC Auto-configuration adds the following on top of Spring's defaults.

1. ContentNegotiatingViewResolver bean

2. Use of ConfigurableWebBindingInitializer

3. Favicon support

4. Registration of MessageCodesResolver

Correct Answer: 1,2,3,4

Explanation: "The auto-configuration adds the following features on top of Spring's defaults:

Inclusion of ContentNegotiatingViewResolver and BeanNameViewResolver beans.

Support for serving static resources, including support for WebJars.

Automatic registration of Converter, GenericConverter, and Formatter beans.

Support for HttpMessageConverters.

Automatic registration of MessageCodesResolver.

Static index.html support.

Custom Favicon support.

Automatic use of a ConfigurableWebBindingInitializer bean.

2. In Spring Boot, what is the default type of ApplicationContext created if Spring MVC is not present?

1. ClassPathXmlApplicationContext

2. AnnotationConfigApplicationContext

3. FileSystemXmlApplicationContext

4. AnnotationConfigServletWebServerApplication Context

Correct Answer: 2

Explanation: "A SpringApplication attempts to create the right type of ApplicationContext on your behalf. The algorithm used to determine a WebApplicationType is the following:

If Spring MVC is present, an AnnotationConfigServletWebServerApplication Context is used

If Spring MVC is not present and Spring WebFlux is present, an AnnotationConfigReactiveWebServerApplication

Context is used

Otherwise, AnnotationConfigApplicationContext is used

3. Which of the following affect which packages get component scanned?

1. Location of @ComponentScan annotated class

2. basePackageClasses attribute of the @ ComponentScan annotation

3. scanBasePackageClasses attribute of @ SpringBootApplication class

4. Configuration property spring.scan.location

Correct Answer: 1, 2, 3

Explanation: @ComponentScan.basePackages specifies the packages to be scanned as a String. basePackageClasses is the type-safe alternative to this (scans the package that each class is in).

SpringBootApplication provides alias fields for both of these methods.

4. In Spring Boot, which of the following ways can be used to customize an auto-configuration class provided in a Spring starter?

1. Override the bean definition (overriding needs to be enabled first)

2. Remove the class

3. Annotate using @Disabled

4. It cannot be customized.

Correct Answer: 1

Explanation: "The simplest way to customize Spring auto-configuration is by changing any property values used by the related beans.

The next option is to create a spring bean to replace a bean created by the auto-configuration. To replace the auto-configuration bean, the new bean needs to have a matching type and/or name (depending on the @Conditional annotation(s) annotating the bean method in the auto-configuration).

Note that in order to be able to override bean definitions using beans with the same name as the original bean, the following application property needs to be set to true in the Spring

Boot application: spring.main.allow-bean-definition-overriding=true

Another option is to disable one or more auto-configuration classes altogether.
This can be accomplished either in the @ EnableAutoConfiguration annotation, by specifying the class(es) or fully qualified class name(s), or by using the spring.autoconfigure.exclude property

with one or more fully qualified class name(s). Removing the class is not recommended and usually not possible if the class is provided in a starter without removing the entire starter (which will remove many of the needed classes)."

5. Which of the following are uses for the spring.factories file?

1. Register application event listeners

2. Register Application-specific Factory classes

3. Locate auto-configuration candidates

4. Bean overriding

Correct Answer: 1,3

Explanation: "The spring.factories file can be used to:

Register application event listeners regardless of how the Spring Boot application is created

(configured).

Implement a class that inherits from SpringApplicationEvent and register it in the

spring.factories file.

Locate auto-configuration candidates in, for instance, your own starter module.

Register a filter to limit the auto-configuration classes considered.

See AutoConfigurationImportFilter.

Activate application listeners that creates a file containing the application process id and/or

creates file(s) containing the port number(s) used by the running web server (if any).

These listeners, ApplicationPidFileWriter and WebServerPortFileWriter, both implement the

ApplicationListener interface.

Register failure analyzers.

Failure analyzers implement the FailureAnalyzer interface and can be registered in the

spring.factories file.

Customize the environment or application context prior to the Spring Boot application has

started up.

Classes that implementing the ApplicationListener, ApplicationContextListener or the

EnvironmentPostProcessor interfaces may be registered in the spring.factories file.

Register the availability of view template providers.

See the TemplateAvailabilityProvider interface.

6. Which of the following affect which packages get component scanned?

1. <context:component-scan> in XML Configuration

2. Configuration property spring.scan.location

3. scanBasePackages attribute of @SpringBootApplication class

4. location field of ComponentScanner bean

Correct Answer: 1,3

Explanation: "@ComponentScan.basePackages specifies the packages to be scanned as a String. basePackageClasses is the type-safe alternative to this (scans the package that each class is in).

SpringBootApplication provides alias fields for both of these methods.

7. In Spring Boot, which of the following ways can be used to customize an auto-configuration class provided in a Spring starter?

1. Override it in a @Configuration

2. Override the bean definition (overriding needs to be enabled first)

3. Make sure the condition specified by its @ ConditionalOn* annotation is false

4. Exclude it in the @EnableAutoConfiguration annotation

Correct Answer: 2, 3, 4

Explanation: "The simplest way to customize Spring auto-configuration is by changing any property values used by the related beans.

The next option is to create a Spring bean to replace a bean created by the auto-configuration. To replace the auto-configuration bean, the new bean needs to have a matching type and/ or name (depending on the @Conditional annotation(s) annotating the bean method in the auto- configuration). Note that in order to be able to override bean definitions using beans with the same name as the original bean, the following application property needs to be set to true in the Spring

Boot application: spring.main.allow-bean-definition-overriding=true

Another option is to disable one or more auto-configuration classes altogether.
This can be accomplished either in the @ EnableAutoConfiguration annotation, by specifying

the class(es) or fully qualified class name(s), or by using the spring.autoconfigure.exclude property with one or more fully qualified class name(s). Removing the class is not recommended and usually not possible if the class is provided in a starter without removing the entire starter (which will remove many of the needed classes)."

8. Spring _____ is referred to as opinionated, using standard industry guidelines as sensible defaults.

1. Boot

2. Security

3. Rest

4. Web

Correct Answer: 1

Explanation: "Spring Boot is Opinionated and uses defaults to enable a quick startup for new projects.

9. In Spring Boot, how can specific auto-configuration classes be disabled?

1. "exclude" attribute of the @SpringBootApplication annotation

2. @Disabled annotationm on the classes

3. Adding explicit configuration classes

4. @ManualConfiguration annotation

Correct Answer: 1, 3

Explanation: "If the class is not on the classpath, you can use the excludeName attribute of the annotation and specify the fully qualified name instead. If you prefer to use @EnableAutoConfiguration rather than @SpringBootApplication, exclude and excludeName are also available. Finally, you can also control the list of auto-configuration classes to exclude by using the spring.autoconfigure.exclude property.

10. Which of the following annotations is used on custom Spring Boot Auto-configuration classes?

1. @ConfigurationCondition

2. @AutoConfiguration

3. @ConditionalConfiguration

4. @Configuration

Correct Answer: 4

Explanation: "Custom auto-configuration classes are simply @Configuration classes using @ConditionalOn* annotations to select the situation in which the Configuration should come into effect.

The other annotations don't exist.

11. Spring Boot provides a default configuration for the log implementation _____ ?

1. Commons Logging

2. Log4J2

3. SLF4J

4. Logback

Correct Answer: 2, 4

Explanation: "Spring Boot uses Commons Logging for all internal logging but leaves the underlying log implementation open. Default configurations are provided for Java Util Logging, Log4J2, and Logback. In each case, loggers are pre-configured to use console output with optional file output also available.

12. The annotation @SpringBootApplication is a shorthand for the following annotations.

1. @Configuration

2. @EnableAutoConfiguration

3. @Application

4. @ComponentScan

Correct Answer: 1, 2, 4

Explanation: A single @SpringBootApplication annotation can be used to enable those three features, that is:

@EnableAutoConfiguration: enable Spring Boot's auto-configuration mechanism

@ComponentScan: enable @Component scan on the package where the application is located (see the best practices)

@Configuration: allow to register extra beans in the context or import additional configuration classes

13. Which of the following are condition annotations used by Spring Boot for auto-configuration?

1. @ConditionalOnClass

2. @ConditionalOnMissingBean

3. @ConditionalOnMVC

4. @ConditionalOnJava

Correct Answer: 1, 2, 4

Explanation: Here are all of the possible auto-configuration condition annotations:

@ConditionalOnClass - Presence of class on

classpath.

@ConditionalOnMissingClass - Absence of class on classpath.

@ConditionalOnBean - Presence of Spring bean or bean type (class).

@ConditionalOnMissingBean - Absence of Spring bean or bean type (class).

@ConditionalOnProperty - Presence of Spring environment property.

@ConditionalOnResource - Presence of resource such as file.

@ConditionalOnWebApplication - If the application is considered to be a web application, that is uses the Spring WebApplicationContext, defines a session scope or has a StandardServletEnvironment.

@ConditionalOnNotWebApplication - If the application is not considered to be a web application.

@ConditionalOnExpression - Bean or configuration active based on the evaluation of a SpEL expression.

@ConditionalOnCloudPlatform - If specified cloud platform, Cloud Foundry, Heroku or SAP, is active.

@ConditionalOnEnabledEndpoint - Specified endpoint is enabled.

@ConditionalOnEnabledHealthIndicator - Named health indicator is enabled.

@ConditionalOnEnabledInfoContributor - Named info contributor is enabled.

@ConditionalOnEnabledResourceChain - Spring resource handling chain is enabled.

@ConditionalOnInitializedRestarter - Spring DevTools RestartInitializer has been applied with non-null URLs.

@ConditionalOnJava - Presence of a JVM of a certain version or within Condition Annotation Condition Factor a version range.

@ConditionalOnJndi - Availability of JNDI InitialContext and specified JNDI locations exist.

@ConditionalOnManagementPort - Spring Boot Actuator management port is either: Different from server port, same as server port or disabled.

@ConditionalOnRepositoryType - Specified type of Spring Data repository has been enabled.

@ConditionalOnSingleCandidate - Spring bean of specified type (class) contained in bean factory and single candidate can be determined.

14. In Spring Boot, in order to use YAML to configure application properties, you must add _____.

1. SnakeYAML library to the classpath

2. @EnableYaml annotation

3. YamlPropertySourceLoader bean

4. Nothing, since it is available by default

Correct Answer: 4

Explanation: "The SpringApplication class automatically supports YAML as an alternative to properties whenever you have the SnakeYAML library on your classpath.

If you use "Starters" SnakeYAML is automatically provided by spring-boot-starter.

15. Spring MVC Auto-configuration adds the following on top of Spring's defaults?

1. ContentNegotiatingViewResolver bean

2. Use of ConfigurableWebBindingInitializer

3. Favicon support

4. Registration of MessageCodesResolver

Correct Answer: 1, 2, 3, 4

Explanation: "The auto-configuration adds the following features on top of Spring's defaults:

Inclusion of ContentNegotiatingViewResolver and BeanNameViewResolver beans.

Support for serving static resources, including support for WebJars.

Automatic registration of Converter, GenericConverter, and Formatter beans.

Support for HttpMessageConverters.

Automatic registration of MessageCodesResolver..

Static index.html support.

Custom Favicon support.

Automatic use of a ConfigurableWebBindingInitializer bean.

16. Which of the following affect what Spring Boot auto-configures?

1. System performance

2. Declared Beans

3. Other running Spring Applications

4. Dependencies in Classpath

Correct Answer: 2, 4

Explanation: "Spring Boot auto-configuration attempts to automatically configure your Spring application based on the jar dependencies that you have added. For example, if HSQLDB is on your classpath, and you have not manually configured any database connection beans, then Spring Boot auto-configures an in-memory database.

Some explicitly declared beans will be used based on their name (e.g. a bean named ""transactionManager"" will be used as the application's transaction manager).

Boot Intro

1. What is the name of the default property configuration file in Spring Boot?

1. configuration.properties

2. application.properties

3. config.txt

4. settings.properties

Correct Answer: 2

Explanation: "The default configuration file is application.properties (or application.yml if YAML is used).

2. Which of the following interfaces can be implemented for functionality which runs once per application startup?

1. ApplicationRunner

2. SpringRunner

3. CommandLineRunner

4. StartupRunner

Correct Answer: 1,3

Explanation: "If you need to run some specific code once the SpringApplication has started, you can implement the ApplicationRunner or CommandLineRunner interfaces. Both interfaces work in the same way and offer a single run method, which is called just before SpringApplication.run() completes.

The CommandLineRunner interfaces provides access to application arguments as a string array, whereas the ApplicationRunner uses the ApplicationArguments interface

3. Which of the following are possible build output types when using Spring Boot?

1. Non-executable JAR

2. Fat JAR

3. WAR

4. All of the above.

Correct Answer: 4

Explanation: "Spring Boot offers options of building the application as an executable (fat) JAR, non-executable JAR or WAR.

4. Which of the following are advantages offered by Spring Boot?

1. Easy Configuration

2. Less boiler plate code

3. It can be run from a command line.

4. Simple Dependency management

Correct Answer: 1,2,3,4

Explanation: All of these are advantages offered by Spring Boot. Spring Boot is designed to be user friendly, easy to use and with less boilerplate code than the alternatives.

5. Which of the following are ways to change the behavior of a Spring Boot application in a certain profile?

1. @Profile annotation

2. application-{profile}.properties

3. application-{profile}.yml

4. @PropertySource annotation

Correct Answer: 1,2,3

Explanation: The following ways of changing the behavior based on profiles exist:

@Profile annotations in java code, application-{profile}.properties and its YAML variant.

6. Which of the following is the default

JSON mapping library used by Spring Boot?

1. Jackson

2. Gson

3. SnakeJSON

4. JSON-B

Correct Answer: 1

Explanation: "Spring Boot provides integration with three JSON mapping libraries: Gson, Jackson and JSON-B.

Jackson is the preferred and default library.

7. Which of the following are valid usages of Spring profiles?

```
1. @Profile({exclude=""production""})
@Configuration
public class MyConfig {
  // code
}

2. @Configuration
public class MyConfig {
     @Profile({""dev""})
     @Bean
     public MyService myService(){
     }
}
```

```
3. @Profile({"!production"})
@Configuration
public class MyConfig {
}

4. @Configuration
public class MyConfig {
@Profile({""dev"",""staging""})
@Bean
public MyService myService(){
}
```

Correct Answer: 2, 3, 4

Explanation: "Configuration classes and bean methods can be annotated with the @Profile annotation.

The Profile annotation does not have an ""exclude attribute"". For negative profiles, profile names need to be prefixed with an exclamation mark (!), like in the 3rd answer.

8. What type of file may have the following content format?

spring:

data:

jpa-dialect: H2

server:

port: 8080”

1. YAML Files

2. Properties Files

3. XML Files

4. Java Configuration files

Correct Answer: 1

Explanation: YAML is the only format of the given ones where indentation signifies nesting

9. Which of the following ways of configuring Spring Boot has the highest precedence (overrides the others)?

1. @TestPropertySource

2. Profile-specific application properties

3. OS environment variables

4. Default Properties

Correct Answer: 1

Explanation: @TestPropertySource annotation has the highest precedence out of these answers

because test settings override normal application settings.

10. What logging is used by Spring Boot for internal logging?

1. Commons Logging

2. SLF4J

3. Log4J2

4. Java Util Logging

Correct Answer: 1

Explanation: Spring Boot uses Commons Logging for all internal logging but leaves the underlying log implementation open. Default configurations are provided for Java Util Logging, Log4J2, and Logback. In each case, loggers are pre-configured to use console output with optional file output also available.

11. Which is the default embedded web server used by Spring Boot?

1. JBoss

2. Tomcat

3. Undertow

4. Jetty

Correct Answer: 2

Explanation: "Tomcat is the default embedded web server used by Spring Boot. Other options available are Jetty and UnderTow.

12. Which is the default logging level in Spring Boot?

1. DEBUG

2. INFO

3. WARN

4. ERROR

Correct Answer: 2

Explanation: The default logging level is INFO.

13. From which of these places does Spring Boot draw configuration properties?

1. application-properties.xml

2. Spring Application default properties

3. OS environment variables

4. JNDI attributes from java:comp/env

Correct Answer: 2, 3, 4

Explanation: Spring Boot uses a very particular PropertySource order that is designed to allow sensible overriding of values. Properties are considered in the following order:

Devtools global settings properties in the $HOME/.config/spring-boot directory when devtools is active.

@TestPropertySource annotations on your tests.

Properties attribute on your tests. Available on @SpringBootTest and the test annotations for testing a particular slice of your application.

Command line arguments.

Properties from SPRING_APPLICATION_JSON (inline JSON embedded in an environment variable or system property).

ServletConfig init parameters.

ServletContext init parameters.

JNDI attributes from java:comp/env.

Java System properties (System.getProperties()).

OS environment variables.

A RandomValuePropertySource that has properties only in random.*.

Profile-specific application properties outside of your packaged jar (application-{profile}.properties and YAML variants).

Profile-specific application properties packaged inside your jar (application-{profile}.properties and YAML variants).

Application properties outside of your packaged jar (application.properties and YAML variants).

Application properties packaged inside your jar (application.properties and YAML variants).

@PropertySource annotations on your @ Configuration classes. Please note that such property sources are not added to the Environment until the application context is being refreshed. This is too late to configure certain properties such as logging.* and spring.main.* which are read before refresh begins.

Default properties (specified by setting SpringApplication.setDefaultProperties).

14. What is a JAR file with all needed dependencies needed to run the application called?

1. full JAR

2. application JAR

3. fat JAR

4. dependency JAR

Correct Answer: 3

Explanation: "Executable jars (sometimes called "fat jars" are archives containing your compiled classes along with all of the jar dependencies that your code needs to run.

15. What does the annotation @ ConfigurationProperties("example") do?

1. Uses the application profile "example"

2. Binds external properties from root "example"

3. Populates Application Properties for "example"

4. Validates Configuration Properties against validation group "example"

Correct Answer: 2

Explanation: Using the @Value(""${property}"") annotation to inject configuration properties can sometimes be cumbersome, especially if you are working with multiple properties or your data is hierarchical in nature. Spring Boot provides an alternative method of working with properties that lets strongly typed beans govern and validate the configuration of your application.

It is possible to bind a bean declaring standard JavaBean properties as shown.

Boot Testing

1. Which of the following dependencies are included in the spring-boot-starter-test starter?

1. Mockito

2. JsonPath

3. Cucumber

4. TestNG

Correct Answer: 1, 2

Explanation: "The spring-boot-starter-test starter adds the following test-scoped dependencies to the classpath:

JUnit - Unit-testing framework.

Spring Test and Spring Boot Test

AssertJ - Fluent assertions for Java.

Hamcrest - Framework for writing matchers that are both powerful and easy to read.

Mockito - Mocking framework for Java.

JSONassert - Tools for verifying JSON

representation of data.

JsonPath - A Java DSL for reading JSON documents.

2. How should configuration classes in src/test/java be annotated so that any declared beans are added to the ApplicationContext in tests?

1. @Configuration

2. @TestConfiguration

2. @ConfigurationProperties

4. @TestContext

Correct Answer: 2

Explanation: @TestConfiguration can be used on an inner class of a test to customize the primary configuration. When placed on a top-level class, @TestConfiguration indicates that classes in src/test/java should not be picked up by scanning. You can then import that class.

3. @SpringBootTest is meta annotated with _____?

1. @RunWith(SpringExtension.class)

2. @ExtendWith(SpringExtension.class)

3. @RunWith(SpringRunner.class)

4. @ExtendWith(SpringRunner.class)

Correct Answer: 2

Explanation: @SpringBootTest is meta annotated with @ExtendWith(SpringExtension.class), because it uses JUnit 5. @RunWith(SpringRunner.class) is the annotation used with JUnit 4.

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

1. Mocks all references in the application context to a bean

2. No extra dependencies are required

3. Can be used as a meta annotation

4. Can be used with MockitoJUnitRunner

Correct Answer: 1, 3

Explanation: Both the @MockBean and @Mock annotation can be used to create Mockito mocks but there are

Here are few more differences between the two annotations:

1. @Mock can only be applied to fields and

parameters while @MockBean can only be applied to classes and fields.

2. @Mock can be used to mock any Java class or interface while @MockBean only allows for

mocking of Spring beans or creation of mock Spring beans.

3. @MockBean can be used to mock existing beans but also to create new beans that will

belong to the Spring application context.

4. To be able to use the @MockBean annotation, the Spring runner

(@RunWith(SpringRunner.class)) has to be used to run the associated test.

@MockBean can be used to create custom annotations for specific, reoccurring, needs

When running Spring Boot Tests, both @Mock and @MockBean are included in the spring-boot-starter-test.

5. The annotation _____ is used on test-classes testing only Spring MVC components.

1. @WebTest

2. @MvcTest

3. @Test

4. @WebMvcTest

Correct Answer: 4

Explanation: @WebMvcTest is an annotation that can be used in combination with @ RunWith(SpringRunner.class) for a typical Spring MVC test. Can be used when a test focuses only on Spring MVC components. Using this annotation will disable full auto-configuration and ins”

6. The annotation _____ is used on test-classes only containing JPA components?

1. @JpaTest

2. @DataTest

3. @DataJpaTest

4. @Test

Correct Answer: 3

Explanation: “@DataJpaTest is an annotation that can be used in combination with @ RunWith(SpringRunner.class) for a typical JPA test. Can be used when a test focuses only on JPA components. Using this annotation will disable full auto-configuration”

7. Which of the following are advantages of running web application tests on a random port?

1. Parallelization

2. Security

3. Integration testing

4. Debugging capabilities

Correct Answer: 1

Explanation: The advantage when running web application tests on a random port is that it will not clash with other instances of the application (such as a running application or other tests)

8. Which of these annotations can be used to insert a Mock of a Spring bean into the application context?

1. @Mock

2. @MockBean

3. @InjectMocks

4. @InjectMockBean

Correct Answer: 2

Explanation: @MockBean can be used to inject a

Mock instead of a Spring Bean into the Application Context.

9. Which of the following are auto-configured when using @WebMvcTest?

1. Transactions

2. Caching

3. Jackson

4. Templating engine

Correct Answer: 2, 3, 4

Explanation: To test whether Spring MVC controllers are working as expected, use the @WebMvcTest annotation. @WebMvcTest auto-configures the Spring MVC infrastructure and limits scanned beans to @Controller, @ControllerAdvice, @JsonComponent, Converter, GenericConverter, Filter, HandlerInterceptor, WebMvcConfigurer, and HandlerMethodArgumentResolver.

Regular @Component and @ConfigurationProperties beans are not scanned when the @WebMvcTest annotation is used. @EnableConfigurationProperties can be used to include @ConfigurationProperties beans.

Often, @WebMvcTest is limited to a single controller and is used in combination with @MockBean to provide mock implementations for

required collaborators.

@WebMvcTest also auto-configures MockMvc. Mock MVC offers a powerful way to quickly test MVC controllers without needing to start a full HTTP server.

10. Which of the following are web environment options offered by @ SpringBootTest?

1. FULL

2. MOCK

3. NONE

4. STANDALONE

Correct Answer: 2,3

Explanation: "Four different types of web environments can be specified using the webEnvironment attribute of the @SpringBootTest annotation:

MOCK - Loads a web ApplicationContext and provides a mock web environment. Does not start a web server.

RANDOM_PORT - Loads a WebServerApplicationContext, provides a real web environment and starts an embedded web server listening on a random port. The port allocated

can be obtained using the @LocalServerPort annotation or @Value(“”${local.server.port}””). Web server runs in a separate thread and server-side transactions will not be rolled back in transactional tests.

DEFINED_PORT - Loads a WebServerApplicationContext, provides a real web environment and starts an embedded web server listening on the port configured in the application properties, or port 8080 if no such configuration exists. Web server runs in a separate thread and server-side transactions will not be rolled back in

transactional tests.

NONE - Loads an ApplicationContext without providing any web environment.

11. Which of the following are auto-configured when using @DataJpaTest?

1. JdbcTemplate

2. Caching

3. JSR-303 bean validation

4. Liquibase and Flyway

Correct Answer: 1, 2, 4

Explanation: The @DataJpaTest annotation auto-configures the following:

1. Caching

2. Spring Data JPA repositories

3. Flyway database migration tool

4. A DataSource - The data-source will, as default, use an embedded in-memory database (test database).

5. Data source transaction manager - A transaction manager for a single DataSource.

6. A JdbcTemplate

7. Liquibase database migration tool

8. JPA base configuration for Hibernate

9. Spring transaction

10. A test database

11. A JPA entity manager for tests

12. What does the annotation @ SpringBootTest do?

1. Searches for a @SpringBootConfiguration

2. Registers a TestRestTemplate and/or WebTestClient bean

3. Creates a web environment for the test

4. All of the above.

Correct Answer: 4

Explanation: "Annotation that can be specified on a test class that runs Spring Boot based tests. Provides the following features over and above the regular Spring TestContext Framework:

Uses SpringBootContextLoader as the default ContextLoader when no specific @ ContextConfiguration(loader=...) is defined.

Automatically searches for a @ SpringBootConfiguration when nested @ Configuration is not used, and no explicit classes are specified.

Allows custom Environment properties to be defined using the properties attribute.

Allows application arguments to be defined using the args attribute.

Provides support for different webEnvironment modes, including the ability to start a fully running web server listening on a defined or random port.

Registers a TestRestTemplate and/or WebTestClient bean for use in web tests that are using a fully running web server.

13. Which of the following are valid ways to run an MVC test on a random port?

1. Using the @RandomPort annotation

2. Setting the property server.port=0

3. Using the webEnvironment=RANDOM_PORT attribute of @SpringBootTest

4. Using the port=RANDOM attribute of @ SpringBootTest

Correct Answer: 2, 3

Explanation: @SpringBootTest provides support for different web environment modes to create for the test using the webEnvironment element of the @SpringBootTest annotation. The following web environment modes are available: DEFINED_PORT (creates a web application con.

14. Which of the following classes is created and configured when the annotation @SpringBootTest is used with the attribute webEnvironment set to RANDOM_PORT or DEFINED_PORT?

1. TestRestTemplate

2. MockMvc

3. SpringApplication

4. TestPropertySource

Correct Answer: 1

Explanation: "Alternatively, if you use the @SpringBootTest annotation with WebEnvironment.RANDOM_PORT or WebEnvironment.DEFINED_PORT, you can inject a fully configured TestRestTemplate and start using it. If necessary, additional customizations can be applied through the RestTemplateBuilder bean. Any URLs that do not specify a host and port automatically connect to the embedded server.

15. Which of the following are auto-configured when using @WebMvcTest?

1. Thymeleaf

2. Components

3. DataSource

4. MockMvc

Correct Answer: 1, 4

Explanation: "To test whether Spring MVC controllers are working as expected, use the @WebMvcTest annotation. @WebMvcTest auto-configures the Spring MVC infrastructure and limits scanned beans to @Controller, @ControllerAdvice, @JsonComponent, Converter, GenericConverter, Filter, HandlerInterceptor, WebMvcConfigurer, and HandlerMethodArgumentResolver. Regular @Component and @ConfigurationProperties beans are not scanned when the @WebMvcTest annotation is used. @EnableConfigurationProperties can be used to

include @ConfigurationProperties beans.

Often, @WebMvcTest is limited to a single controller and is used in combination with @MockBean to provide mock implementations for required collaborators.

@WebMvcTest also auto-configures MockMvc. Mock MVC offers a powerful way to quickly test MVC controllers without needing to start a full HTTP server.

Container

1. Which of the following are limitations of CGLIB Proxies?

1. Requires a third-party library

2. Requires the class to be non-final

3. Requires the methods to be non-final

4. Requires an interface to be declared

Correct Answer: 1, 2, 3

Explanation: Spring AOP can also use CGLIB proxies. This is necessary to proxy classes rather than interfaces.

Limitations of CGLIB proxies are:

Requires the class of the proxied object to be non-final.

Subclasses cannot be created from final classes.

Requires methods in the proxied object to be non-final.

Final methods cannot be overridden.

Does not support self-invocations.

Self-invocation is where one method of the object invokes another method on the same

object.

Requires a third-party library.

Not built into the Java language and thus require a library. The CGLIB library has been

Included into Spring, so when using the Spring framework, no additional library is required.

2. Where can the @Lazy annotation be placed to cause Spring beans to be instantiated lazily?

1. Methods annotated with @Bean

2. Classes annotated with @Configuration

3. Classes annotated with @Component or another stereotype annotation

4. Test Classes

Correct Answer: 1, 2, 3, 4

Explanation: "Methods annotated with the @Bean annotation.

Bean will be lazy or not as specified by the boolean parameter to the @Lazy annotation

(default value is true).

Classes annotated with the @Configuration annotation.

All beans declared in the configuration class will be lazy or not as specified by the boolean

parameter to the @Lazy annotation (default value is true).

Classes annotated with @Component or any related stereotype annotation.

The bean created from the component class will be lazy or not as specified by the boolean parameter to the @Lazy annotation (default value is true)."

3. What is the maximum number of profiles that can be active at the same time?

1. 1

2. 2

3. 3

4. None of the above.

Correct Answer: 4

Explanation: "There does not seem to be any limitation concerning how many profiles that can

be used in a Spring

application. The Spring framework (in the class ActiveProfilesUtils) use an integer to iterate over an

array of active profiles, which implies a maximum number of 2^32 - 1 profiles."

4. Which of the following statements are true regarding the Spring Application Context?

1. It instantiates Spring Beans

2. It is the front servlet of the application.

3. It resolves external configuration properties.

4. A running application has only one.

Correct Answer: 1,3

Explanation: "The Front Servlet of the application is the DispatcherServlet. It is possible for applications to have multiple applications context, arranged in a parent-child hierarchy.

5. If needed, Spring application contexts can be bootstrapped in the following situations.

1. JUnit tests

2. A second instance is required

2. Web applications

4. Standalone applications

Correct Answer: 1, 3, 4

Explanation: "There are different implementations of the ApplicationContext interface, for each purpose (they can be viewed at the link below).

Spring does not support running multiple instances on the same machine.

6. Which class is responsible for injecting values into beans when using the @Value annotation?

1. BeanFactoryPostProcessor

2. PropertySourcesPlaceholderConfigurer

3. ValuePopulator

4. BeanValuePostProcessor

Correct Answer: 2

Explanation:
"PropertySourcesPlaceholderConfigurer is a specialization of PlaceholderConfigurerSupport that resolves ${...} placeholders within bean definition property values and @Value annotations against the current Spring Environment and its set of PropertySources.

7. Which of the following are advantages of using interfaces in Java?

1. Simple type names

2. Interchangeable implementation classes

3. Better exception handling

4. Increased Modularity

Correct Answer: 2, 4

Explanation: "Interfaces do not help with exception handling, nor do they necessarily have simpler names. Their advantages include being able to change the implementation without needing to change classes that depend on it and, as a result, increased modularity

8. By default, the @ComponentScan class scans in the following locations.

1. The entire Spring application

2. The entire file system

3. The current package

4. Nowhere. The path must be specified.

Correct Answer: 3

9. Which of the following are ways to call a method after initializing a Spring Bean?

1. afterPropertiesSet() defined in InitializingBean

2. init-method XML attribute

3. call in constructor

4. @PostConstruct

Correct Answer: 1,2,4

Explanation: Calling in the constructor would be before initializing the class

10. What is the purpose of the interface MessageSource?

1. Mapping error codes to messages

2. Storing HTTP requests

3. Store of common messages

4. Internationalization

Correct Answer: 4

Explanation: "MessageSource is a strategy interface for resolving messages, with support for the parameterization and internationalization of such messages.

11. What is the name of objects created by the Spring IoC container?

1. POJOs

2. Beans

3. Spring Components

4. Spring Objects

Correct Answer: 2

Explanation: "In Spring, the objects that form the backbone of your application and that are managed by the Spring IoC container are called beans. A bean is an object that is instantiated, assembled, and managed by a Spring IoC container. Otherwise, a bean is simply one of many objects in your application. Beans, and the dependencies among them, are reflected in the configuration metadata used by a container.

12. What do SpEL expressions starting with $ reference?

1. Properties in the application environment

2. Spring Beans

3. Literal Values

4. JVM Properties

Correct Answer: 1

Explanation: Expressions starting with $.

Such expressions reference a property name in the application's environment. These

Expressions are evaluated by the PropertySourcesPlaceholderConfigurer Spring bean prior

to bean creation and can only be used in @Value annnotations.

13. What does the prototype bean scope do in Spring Framework?

1. It autowires the appropriate Bean class based on the given prototype argument

2. It creates a new bean instance every time a request for that specific bean is made

3. It creates a single instance to be used for all requests.

4. It picks the right class out of a class hierarchy

Correct Answer: 2

Explanation: "Prototype bean creates a new bean instance for every usage

14. Which of the following is considered best practice?

1. @ComponentScan({“org”, “com”})

2. @ComponentScan({com.example.app})

3. @ComponentScan(“com”)

4. @ComponentScan(“com.example.app”)

Correct Answer: 4

Explanation: It is recommended to add the minimal amount of packages to ComponentScan so that it is quicker and also so that only components in the application, that should be scanned.

You may notice that option 2 and 4 are quite same but the problem with the second option is that there are no quotes.”

15. Which of the following are implementations of the ApplicationContext interface?

1. ClassPathXmlApplicationContext

2. BeanFactoryApplicationContext

3. FileSystemXmlApplicationContext

4. YamlApplicationContext

Correct Answer: 1, 3

Explanation: "BeanFactoryApplicationContext does not exist (ApplicationContext already extends BeanFactory).

16. Which of the following annotations will instruct the IoC Container to instantiate the beans declared in the Demo1 and Demo2 configuration classes?

1. @Import({Demo1.class, Demo2.class})

2. @Imports(Demo1.class, Demo2.class)

3. @Include({Demo1.class, Demo2.class})

4. @DependsOn(Demo1.class, Demo2.class)

Correct Answer: 1

Explanation: "@Import indicates one or more component classes to import â€" typically @ Configuration classes.

Multiple values are added to an annotation attribute using {}, as arrays.

17. Which of the following are valid ways of defining Spring bean metadata?

1. XML files

2. Java annotations

3. YAML Files

4. Java code

Correct Answer: 1, 2, 4

Explanation: "YAML Files do not directly define bean metadata.

18. Which of the following annotations can be used to specify which bean to inject?

1. @Named

2. @Primary

3. @Qualifier

4. @BeanName

Correct Answer: 1,2

Explanation: @Primary is an effective way to use auto-wiring by type with several instances when one primary candidate can be determined. When you need more control over the selection process, you can use Spring's @Qualifier annotation. You can associate qualifier va

19. Which of the following methods will be called last in the bean lifecycle?

1. destroy method in the DisposableBean interface

2. destroy-method as specified in the Spring XML Configuration

3. Any methods annotated with @PreDestroy

4. Any method named “destroy”

Correct Answer: 3

Explanation: “When the Spring application context is to shut down, the beans in it will receive destruction

callbacks in this order:

Any methods in the bean implementation class annotated with @PreDestroy are invoked.

Any destroy method in a bean implementation class implementing the

DisposableBean interface is invoked.

If the same destruction method has already been invoked, it will not be invoked again.

Any custom bean destruction method is invoked.

Bean destruction methods can be specified either in the value of the destroy-method

attribute in the corresponding <bean> element in a Spring XML configuration or in

the destroyMethod property of the @Bean annotation.

If the same destruction method has already been invoked, it will not be invoked

again."

20. Which of the following methods can be used to shutdown a Spring Boot application?

1. SpringBootApplication.shutdown()

2. AbstractApplicationContext.close()

3. SpringApplication.exit()

4. AbstractApplicationContext.
registerShutdownHook()

Correct Answer: 2, 3, 4

Explanation: "SpringBootApplication is an annotation, so it does not have methods.

21. Which of the following annotations can be used to give precedence to a particular bean?

1. @Order

2. @MainBean

3. @First

4. @Primary

Correct Answer: 4

Explanation: Because auto wiring by type may lead to multiple candidates, it is often necessary to have more control over the selection process. One way to accomplish this is with Spring's @Primary annotation. @Primary indicates that a particular bean should be given preference when multiple beans are candidates to be autowired to a single-valued dependency. If exactly one primary bean exists among the candidates, it becomes the autowired value.

22. Which of the following annotations are used on classes which contain methods defining beans?

1. @Component

2. @Autowired

3. @Bean

4. @Configuration

Correct Answer: 4

Explanation: In Java based configuration, instead of writing bean configurations inside a xml file, we choose to write bean definitions inside a class. That class is annotated with @Configuration, which means that particular class contains bean definitions in form of methods with @Bean annotation.

Example:

```
@Configuration
public class AppConfiguration {
@Bean
public Student getStudent() {
 return new Student();
}
}
```

23. Which of the following statements is true regarding the code below?

```
@Profile(""prod"")
@Configuration
public class MyConfig{
     @Bean
     public DataSource dataSource(){ ..."
```

1. It is invalid.

2. The @Profile annotation can only be applied to methods.

3. The Spring Actuator monitoring profile "prod" will be used for data access performance measuring.

4. This code will not have any effect. @ ContextConfiguration should be used instead of @ Configuration.

5. A DataSource bean will be created if the profile is

“prod”.

Correct Answer: 4

Explanation: “The sample code is valid. If ““prod”” is among the current active profiles, a bean named ““dataSource”” will be instantiated by executing the code in the method.

24. Are beans lazily or eagerly instantiated by default?

1. Lazy

2. Eager

3. Depends on the bean

4. Neither

Correct Answer: 2

Explanation: “By default, ApplicationContext implementations eagerly create and configure all singleton beans as part of the initialization process. Generally, this pre-instantiation is desirable, because errors in the configuration or surrounding environment are discovered immediately.

When this behavior is not desirable, you can prevent pre-instantiation of a singleton bean by marking the bean definition as being lazy-initialized. A lazy-initialized bean tells the IoC container to create a bean instance when it is first

requested, rather than at startup.

25. The @PropertySource annotation is used for _____?

1. Binding a properties file to a POJO

2. Specifying the location of application.properties files

3. Using Java configuration for application.properties

4. Declaring the path to the source files

Correct Answer: 1

Explanation: "The @PropertySource annotation provides a convenient and declarative mechanism for adding a PropertySource to Spring's Environment.

See the following example:

```
@Configuration
@PropertySource(""classpath:/com/myco/app.
properties"")
public class AppConfig {

    @Autowired
    Environment env;

    @Bean
    public TestBean testBean() {
```

```
        TestBean testBean = new TestBean();
        testBean.setName(env.
getProperty(“”testbean.name””));
        return testBean;
    }
}
```

26. What do SpEL expressions starting with # reference?

1. Properties in the application environment

2. Spring Beans

3. Literal Values

4. JVM Properties

Correct Answer: 3

Explanation: “Expressions starting with # are literal values. Spring Expression Language expressions parsed by a SpEL expression parser and evaluated by a SpEL expression instance.

27. Which of the following are advantages of Dependency Injection?

1. Reduces number of dependencies

2. Loose coupling

3. Interface segregation

4. Reduce Boilerplate code

Correct Answer: 2, 4

Explanation: Dependency Injection does not reduce the number of dependencies that is the developer's responsibility.

Interface segregation also has no connection.

DI reduces boilerplate code required to inject dependencies and makes classes less loosely coupled (as this boilerplate code is not needed)"

28. What is the default bean scope in Spring Framework?

1. prototype

2. application

3. singleton

4. request

Correct Answer: 3

Explanation: "Singleton is the default bean scope in Spring. Only one shared instance of a singleton bean is managed, and all requests for beans with an ID or IDs that match that bean definition result in that one specific bean instance being returned by the Spring container.

29. Given the annotation @Profile({“p1”, “!p2”}), which of the following combinations of profiles will fulfill the condition?

1. p1,p2

2. p1

3. p2

4. p3

Correct Answer: 2, 4

Explanation: If a given profile is prefixed with the NOT operator (!), the annotated component will be registered if the profile is not active, for example, given @Profile({””p1””, ””!p2””}), registration will occur if profile ‘p1’ is active or if profile ‘p2’ is not active.

30. A bean instance in _____ scope lives within the lifetime of a single HTTP Request?

1. global session

2. request

3. session

4. application

Correct Answer: 2

Explanation: "A Bean is created when there is a request and it is destroyed after the request is completed.

31. Which of the following are valid targets for the @Profile annotation?

1. bean classes

2. any method

3. bean methods

4. bean fields

Correct Answer: 1,3

Explanation: "The @Profile annotation may be used in any of the following ways:

* as a type-level annotation on any class directly or indirectly annotated with @Component, including @Configuration classes

* as a meta-annotation, for the purpose of composing custom stereotype annotations

* as a method-level annotation on any @Bean method

32. Which of the following annotations can be used to change the order in which the

IoC Container instantiates beans?

1. @Order

2. @Lazy

3. @Import

4. @DependsOn

Correct Answer: 1,2,3,4

Explanation: All of these annotations affect the order in which the IoC Container instantiates beans.

@Lazy makes the Container only instantiate the annotated bean when it is called.

@Order directly specifies the order in which beans are instantiated.

@Import and @DependsOn both make sure that the annotated beans are instantiated after their dependencies.

33. Why is field-based dependency injection not recommended?

1. Because it uses the Java Reflection API, which may adversely impact performance

2. Because it does not use Class-based Dependency Injection

3. It disallows final/immutable field declaration

4. It hides the dependencies

Correct Answer: 3,4

Explanation: "Concerns regarding Reflection API performance are unfounded, as it is used in Dependency Injection anyways.

34. Which of the following are stereotype annotations?

1. @Bean

2. @Controller

3. @Repository

4. @Service

Correct Answer: 2, 3, 4

Explanation: "Stereotype Annotations are meta-annotations of @Component.

Bean annotation is used on methods defining a bean inside a @Configuration class.

35. When the @Autowired annotation is placed on _____, it will _____

1. a method with multiple arguments, throw an

exception

2. a collection of beans, inject all beans of that type

3. a setter, autowire using the setter

4. a final field, throw an exception

Correct Answer: 2,3,4

Explanation: "@Autowired marks a constructor, field, setter method, or config method as to be autowired by Spring's dependency injection facilities. This is an alternative to the JSR-330 Inject annotation, adding required-vs-optional semantics."

36. Which of the following bean scopes exist?

1. local

2. webapp

3. request

4. prototype

Correct Answer: 3,4

Explanation: "There are six scopes for beans in Spring framework namely singleton, prototype, request, application, session and websocket.

Local and Webapp Bean scopes do not exist

37. All exceptions classes in Spring _____.

1. are checked

2. are unchecked

3. inherit SpringException

4. are either checked or unchecked

Correct Answer: 2

Explanation: Spring prefer Unchecked Exceptions as it gives developers freedom of choice as to decide where to implement error handling and removes coupling related to exceptions. It also removes cluttered code as there is no requirement of try-catch blocks.

38. The annotation _____ instructs the IoC container to prepare the specified bean before initializing the annotated bean definition.

1. @Bean

2. @Configuration

3. @DependsOn

4. @Lazy

Correct Answer: 3

Explanation: "When @DependsOn is used, the IoC Container makes sure that the dependency is available before the annotated bean is instantiated.

39. Which functions are possible only when using interfaces as spring beans?

1. Dependency Injection

2. Exception Handling

3. JDK Dynamic Proxying

4. Test Application Context

Correct Answer: 3

Explanation: "JDK Dynamic Proxies work by proxying interfaces. If the classes do not have interfaces, then only CGLIB Proxies can be used (as these generate subclasses).

40. Classes annotated with _____ will be detected by component scanning.

1. @Component

2. @Configuration

3. @Bean

4. @Scanned

Correct Answer: 1,2

Explanation: "Component scan picks up classes annotated with @Component or stereotype annotations (which are simply meta annotated with @Component) like @Configuration.

@Bean is applied at method level.

The annotation @Scanned does not exist in Spring."

JDBC

1. Which of the following methods in JdbcTemplate are recommended for read only operations?

1. query

2. update

3. execute

4. queryForList

Correct Answer: 1, 4

Explanation: query and queryForList method are recommended for read only operations.

2. JdbcTemplate is an example of the _____ design pattern.

1. Factory

2. Adapter

3. Decorator

4. Template

Correct Answer: 4

Explanation: The name JdbcTemplate derives from the design pattern Template Design Pattern.

3. Which of the following are callback interfaces for the class JdbcTemplate?

1.RowCallbackHandler

2. ResultSetCallback

3. ResultSetExtractor

4. RowMapper

Correct Answer: 1,3,4

Explanation: "RowCallbackHandler, ResultSetExtractor and RowMapper are all callback interfaces for JdbcTemplate. When just mapping the rows to objects, RowMapper is recommended.

4. Which of the following is a callback interface for the class JdbcTemplate?

1. PreparedStatementExtractor

2., ResultSetCallback

3. ResultSetExtractor

4. ResultSetMapper

Correct Answer: 3

Explanation: "ResultSetExtractor is the only class which exists out of these options. It is a callback interface used by JdbcTemplate's query methods. Implementations of this interface perform the actual work of extracting results from a ResultSet

5. Which of the following are purposes of the JdbcTemplate class?

1. Reduces JDBC boilerplate code

2. Proper Exception Handling

3. Simplification of JDBC usage in Spring applications

4. Convenient methods for CRUD operations

Correct Answer: 1,2,3,4

Explanation: "The JdbcTemplate provides all of the above benefits.

6. Which of the following methods in JdbcTemplate is recommended for DDL operations?

1. query

2. update

3. execute

4. queryForList

Correct Answer: 3

Explanation: execute is generally intended for DDL operations

7.What is the root data access exception class in Spring Framework?

1. SpringDataException

2. DataAccessException

3. TransientDataAccessException

4. ConstraintViolationException

Correct Answer: 2

Explanation: "DataAccessException is the root data access exception class in Spring Framework.

8. When does the JdbcTemplate class acquire a database connection?

1. During application startup

2. During JdbcTemplate instantiation

3. When one of its execute method is called

4. None of the above.

Correct Answer: 3

Explanation: "JdbcTemplate acquires a database connection only during a method call, which requires data access, and releases it as soon as the rows are fetched. This is so that resources are not held longer than necessary.

Spring Data JPA

1. Which of the following JPA repository method names are valid and able to generate a Query?

1. findByStartDateBetween

2. findPeopleDistinctByLastnameOrFirstname

3. findByFirstname

4. findAgeLessThan

Correct Answer: 1, 2, 3

Explanation: “findAgeLessThan is invalid because it does not contain the word ““By”” (i.e. findByAgeLessThan would be correct)

2. Which of these annotation is used for specifying a query to be used with a Spring Data JPA Repository method?

1. @SqlQuery

2. @DataQuery

3. @JpaQuery

4. @Query

Correct Answer: 4

Explanation: @Query is the name of the annotation.

3. A JPA Repository finder method, which can generate a valid query, might end with the comparison operator _____.

1. GreaterThan

2. Is

3. Null

4. LessThan

Correct Answer: 1,2,3,4

Explanation: "All of these are valid comparators for finder methods.

4. What is invalid in the following code snippet?

```
public interface PersonRepository extends
JpaRepository<Person, Long> {
  @Query(""select p from Person p where
p.emailAddress = ?1"")
   Person findByEmailAddress(String emailAddress);
}
```

1. The method should be called "findPersonByEmailAddress"

2. The annotation should be @SqlQuery instead of @Query

3. The method Parameter emailAddress should be annotated with @Param.

4. None of the above.

Correct Answer: 4

Explanation: The code snippet is valid as is.

5. Which of these is ordering operator which can be used in custom finder method declaration in Spring Data JPA Repository Interface?

1. Ascending

2. Descending

3. Asc

4. Desc

Correct Answer: 3, 4

Explanation: "The correct ordering operators are Asc and Desc.

6. Which of the following JPA repository

method names are valid and able to generate a Query?

1. findByLastnameNot

2. deleteByRoleId

3. countPersonByLastnameIgnoreCase

4. All of the above.

Correct Answer: 4

Explanation: "All of these are valid method names

7. Which of these is comparison operator which can be used in a finder method declaration in Spring Data JPA Repository Interface, which will also produce a valid database query?

1. GreaterThan

2. Between

3. Like

4. Is

Correct Answer: 1,2,3,4

Explanation: "All of these are valid comparison operators. ""Equals"" is equivalent to ""Equals"".

8. A Class implementing the interface _____ have additional methods to ease paginated access to entitites?

1.CrudRepository

2. PaginatedRepository

3. JpaRepository

4. PagingAndSortingRepository

Correct Answer: 4

Explanation: "The PagingAndSortingRepository abstraction adds additional methods to ease paginated access to entities.

MVC

1. Which annotation can be used to have the request body read and deserialized?

1. @JsonDeserialize

2. @RequestBody

3. @RequestDeserialize

4. @RequestMapping

Correct Answer: 2

Explanation: "@JsonDeserialize is not an annotation in Spring.

@RequestMapping is used to configure controllers and their handler methods.

@RequestDeserialize does not exist.

@RequestBody is used for access to the HTTP request body. Body content is converted to the "

2. In Spring MVC, a controller method (inside a class annotated with @ Controller) annotated with @ RequestMapping may have the return type _____.

1. String

2. int

3. void

4. double

Correct Answer: 1, 3

Explanation: "In Spring MVC, controller methods return type needs to be mapped to a view in order to be valid. Returned String is a view name, whereas void means that a ServletResponse argument has been modified to contain this view or similar. Float and int cannot, however, be mapped to a view.

3. Which of the following may be used as Controller method arguments?

1. WebRequest

2. ServletResponse

3. @IpAddress

4. @RequestBody

Correct Answer: 1,2,4

Explanation: "IpAddress is not a valid annotation.

4. _____ is an annotation equivalent to @Controller, but all controller handler methods assume @ResponseBody semantics by default.

1. @RestController

2. @RestComponent

3. @ResponsiveController

4. @ResponseBodyController

Correct Answer: 1

Explanation: "RestController is a convenience annotation that is itself annotated with @Controller and @ResponseBody.

Types that carry this annotation are treated as controllers where @RequestMapping methods assume @ResponseBody semantics by default.

The other annotations do not exist.

5. _____ is the front controller in MVC based applications DispatcherServlet

1. FrontController

2. Controller

3. FrontDispatcherController

Correct Answer: 1

Explanation: Spring MVC, like many other web frameworks, is designed around the front controller pattern where a central Servlet, the DispatcherServlet, provides a shared algorithm for request processing while actual work is performed by configurable, delegate components. This model is flexible and supports diverse workflows.

Controller is an annotation placed on controller classes.

The other 2 classes do not exist.

6. Which of the following may be used as Controller method arguments?

1. ServletRequest

2. @PathVariable

3. Principal

4. ModelAndView

Correct Answer: 1,2,3

Explanation: ModelAndView can only be used as a Controller method return value.

7. Which of the following template engines does Spring MVC support?

1. Underscore

2. JQuery

3. Mustache

4. Thymeleaf

Correct Answer: 3,4

Explanation: As well as REST web services, you can also use Spring MVC to serve dynamic HTML content. Spring MVC supports a variety of templating technologies, including Thymeleaf, FreeMarker, and JSPs. Also, many other templating engines include their own Spring MVC integrations.

8. Which of the following annotations can be applied to Spring MVC controller method parameters?

1.@RequestParam

2. @RequestBody

3. @RequestHeader

4. @ModelAttribute

Correct Answer: 1,2,3,4

Explanation: "@RequestParam: For access to Servlet request parameters. Parameter values are converted to the declared method argument type.

@RequestHeader : For access to request headers. Header values are converted to the declared method argument type.

9. “In the code below, which of the method parameters is most likely to be annotated with @RequestParam?

```
@RequestMapping(“”/{p1}/{p2}””)
public String someHandlerMethod(String p1, String
p2, String p3, String p4) {“
}
```

1. p1

2. p2

3. p3

4. p4

Correct Answer: 3,4

Explanation: “Both p1 and p2 appear in the URI template of the handler method. Because of that, they should be annotated with @PathVariable.

@RequestParam extracts the annotated parameter from the request parameters (appended after? at the end of the URL).

10. Which of the following statements is true about Spring MVC?

1. It provides additional spring bean scopes.

2. Spring MVC is included in the spring core jar, so no additional jar is needed.

3. It has a JSP tag library that enables data binding and themes.

4. Objects do not need to inherit framework specific classes/interfaces.

Correct Answer: 1,3,4

Explanation: "The Spring MVC dependency is not included in Spring Core.

The following bean scopes are added by Spring MVC: session, request, application, websocket.

11. How will adding the @RequestMapping annotation to a @Controller class affect handler methods inside the class?

1. Its path will prefix the methods' paths

2. It will be overwritten by their @RequestMapping annotation.

3. It will override any handler method annotations

4. It will narrow down the handler methods

mapped HTTP verbs.

Correct Answer: 1, 4

Explanation: "The attributes of the annotation applied at class level will be merged with the handler method annotations.

The path of the methods will be prefixed with the class mapping path. The HTTP Verbs will be narrowed down by the class annotation.

12. "In the code below, which of the method parameters is most likely to be annotated with @PathVariable?

```
@RequestMapping(""/{p1}/{p2}"")
public String someHandlerMethod(String p1, String
p2, String p3, String p4) {
}
```

1. p1

2. p2

3. p3

4. p4

Correct Answer: 1, 2

Explanation: "Both p1 and p2 appear in the URI

template of the handler method. Because of that, they should be annotated with @PathVariable.

@RequestParam extracts the annotated parameter from the request parameters (appended after ? at the end of the URL).

13. In Spring MVC, which of the following may be used as Controller return values by default?

1. ServletResponse

2. void

3. Model

4. ModelAndView

Correct Answer: 2, 3, 4

Explanation: "ServletResponse should not be returned by the method, instead it should be added as an argument and written to inside the method.

REST

1. By default, Spring provides HttpMessageConverter for the following message types.

1. Byte arrays

2. SOAP

3. JAXB

4. JSON

Correct Answer: 1, 3, 4

Explanation: here are many implementations of this interface that performs specific conversions. A few

Examples are:

AtomFeedHttpMessageConverter - Converts to/from Atom feeds.

ByteArrayHttpMessageConverter - Converts to/from byte arrays.

FormHttpMessageConverter - Converts to/from HTML forms.

Jaxb2RootElementHttpMessageConverter - Reads classes annotated with the JAXB2 annotations @XmlRootElement and @XmlType and writes classes annotated with @XmlRootElement.

MappingJackson2HttpMessageConverter - Converts to/from JSON using Jackson 2.x.

2. Which of the following statements are true regarding the RestTemplate class?

1. getForEntity returns domain entities

2. getForObject takes an ObjectFactory as an argument

3. When using postForObject, it is impossible to check the HTTP response code.

4. The put and delete methods have a void return type

Correct Answer: 3, 4

Explanation: ForEntity methods return ResponseEntity objects. These contain the response status code as well as the object in the payload.

ObjectFactory is not used as an argument in any method of RestTemplate.

ForObject methods do not return the HTTP status code.

put and delete methods return void.

3. _____ is used to map URL Query String Parameters to handler method arguments?

1. @RequestParam

2. @PathVariable

3. @SessionParam

4. None of the above.

Correct Answer: 1

Explanation: When an annotation @RequestParam is used before a variable of a controller method argument, the value of query parameter which is part of URL gets copied into the same variable of the controller method.

4. @RestController, @RequestBody and @ResponseBody are included in the _____ jar dependency.

1. spring-web

2. spring-core

3. spring-web-mvc

4. spring-mvc

Correct Answer: 1

Explanation: "@RestController, @RequestBody, @ResponseBody are present in the spring-web module that has the Maven group id org.springframework and artifact id spring-web.

5. Which of the following are properties of REST?

1. Statelessness

2. Resilience

3. Scalability

4. Loose coupling between classes

Correct Answer: 1, 3

Explanation: "Resilience and Loose coupling are not addressed by REST directly.

6. A HTTP status code of _____ means forbidden.

1. 203

2. 303

3. 403

4. 503

Correct Answer: 3

Explanation: "403 is the status code for forbidden. Note that the first number of a HTTP status code signifies what HTTP series the code is a part of.

4xx signifies a client error (i.e. the client is trying to do something he can't or isn't allowed to).

7. Which HTTP verb can be used in REST in order to update an existing entity?

1. GET

2. PUT

3. POST

4. DELETE

Correct Answer: 2

Explanation: "Function of PUT: Store the representation in the request body as the (new) state of the resource. Subsequent GET access on the resource is expected to return this representation with HTTP status 200 until PATCHed, a different representation is PUT, or the state of the resource is DELETEd.

8. Which of these URL should be used to access the following Controller class? Assume the port no to be 8080.

```
@RequestMapping(“”/greeting””)
public String sayHello (@RequestParam String name,
@RequestParam int id)
{
   return “”Hello””;
}
```

1. http://localhost:8080/greeting?name=John&id=4324

2. http://localhost:8080/greeting/name=John/id=4324

3. http://localhost:8080/greeting/John/4324

4. http://localhost:8080/greeting?name=John?id=4324

Correct Answer: 1

Explanation: “Request parameters are appended to the URL after the question mark (?) symbol and are separated using ampersand (&) symbol.

9. What is the protocol usually used for REST communication?

1. JMX

2. SOAP

3. HTTP

4. UDP

Correct Answer: 3

Explanation: "HTTP is commonly used in REST communication.

10. A HTTP Response Code of _____ means _____?

1. 300, Server Side Error

2. 400, Success

3. 500, Client Side Error

4. 200, Success

Correct Answer: 4

Explanation: "Http Response Status Codes for a request are 1x-informational, 2x- Success, 3x-Redirectional, 4x-Client Side Error, 5x-Server Side Error.

11. Which HTTP verb is safe, and as such will not result in a state change of the resource?

1. GET

2. PUT

3. POST

4. DELETE

Correct Answer: 1

Explanation: "Function of GET: Retrieve a representation of the data in the response body.

The GET method is safe, meaning that applying it to a resource does not result in a state change of the resource (read-only semantics).

12. _____ is used to map parts of request URL to handler method arguments?

1. @RequestParam

2. @PathVariable

3. @ResponseParam

4. @RequestPart

Correct Answer: 2

Explanation: "@PathVariable annotation is used to copy part of URL into local variables of controller methods. @RequestPart refers to part of a multipart request, not part of the request URL.

Security

1. @EnableGlobalMethodSecurity is annotation used in Spring Security to secure which layer?

1. Controller layer

2. DAO layer

3. Service layer

4. Web layer

Correct Answer: 3

Explanation: "From version 2.0 onwards Spring Security has improved support substantially for adding security to your service layer methods. It provides support for JSR-250 annotation security as well as the framework's original @Secured annotation. From 3.0 you can also make use of new expression-based annotations.

2. In Spring Security _____ comes after _____?

1. Authentication, Authorization

2. Principal, Authentication

3. Authorization, Authentication

4. Principal, Authorization

Correct Answer: 3

Explanation: Authentication is process of validating the user who he claims to be. Once the person is authenticated, he is allowed to perform certain actions based on his role, which is authorization.

3. Which of the following are characteristics of Spring Security?

1. It is a cross cutting concern

2. It provides options for both Authentication and Authorization

3. It is based on standard Servlet Filters

4. All of the above.

Correct Answer: 4

Explanation: Spring Security is a cross cutting concern, so it is implemented using Spring AOP.

It provides multiple options for authorization as well as authentication.

It uses standard Java Servlet Filters.

4. Which of the following methods can be used to restrict access to endpoints using

URL patterns?

1. antMatchers

2. match

3. restrict

4. mvcMatchers

Correct Answer: 1, 4

Explanation: "MvcRequestMatcher can be configured using the mvcMatchers or antMatchers method.

5. In Spring Security, what is the name of the class retrieving the authentication information from the database for a given username?

1. UserDetails

2. AuthenticationProvider

3. UserDetailsService

4. Principal

Correct Answer: 3

Explanation: "UserDetailsService is used by DaoAuthenticationProvider for retrieving a username, password, and other attributes for

authenticating with a username and password. Spring Security provides in-memory and JDBC implementations of UserDetailsService.

You can define custom authentication by exposing a custom UserDetailsService as a bean.

6. In Spring Security, what is the name of the class holding the information regarding high level user permissions?

1. PrincipalAuthority

2. Authority

3. GrantedAuthority

4. UserDetails

Correct Answer: 3

Explanation: "A granted authority is an authority that is granted to the principal on the Authentication (i.e. roles, scopes, etc.)

7. Which of the following annotations allow specifying access constraints using Spring Expession Language (SpEL)?

1. @PreAuthorize

2. @PostAuthorize

3. @PreFilter

4. @Secured

Correct Answer: 1, 2, 3

Explanation: "The annotations @PreAuthorize, @PostAuthorize, @PreFilter and @PostFilter may use SpEL expressions.

8. In Spring Security, how can SpEL-based authorization constraint annotations be enabled?

1. "When set to true, the attribute prePostEnabled enables the annotations @PreAuthorize, @PostAuthorize, @PreFilter and @PostFilter, all of which may use SpEL expressions.

2. @EnableGlobalMethodSecurity(securedEnabled = true)

3. @EnableGlobalMethodSecurity(jsr250Enabled = true) 4. They are enabled by default.

Correct Answer: 1

Explanation: "When set to true, the attribute prePostEnabled enables the annotations @PreAuthorize, @PostAuthorize, @PreFilter and @PostFilter, all of which may use SpEL expressions.

9. In Spring Security, what is the name of the Servlet Filter intercepting all the requests sent to an application?

1. FilterChainProxy

2. DelegatingProxy

3. DelegatingFilterProxy

4. SecurityFilterChain

Correct Answer: 3

Explanation: "Spring provides a Filter implementation named DelegatingFilterProxy that allows bridging between the Servlet container's lifecycle and Spring's ApplicationContext. The Servlet container allows registering Filters using its own standards, but it is not aware of Spring defined Beans. DelegatingFilterProxy can be registered via standard Servlet container mechanisms, but delegate all the work to a Spring Bean that implements Filter.

10. In Spring Security, what is the name of the class holding user information such as the username and password before Authentication?

1. AuthenticationManager

2. AuthenticationProvider

3. UserDetails

4. GrantedAuthority

Correct Answer: 3

Explanation: “UserDetails is returned by the UserDetailsService. The DaoAuthenticationProvider validates the UserDetails and then returns an Authentication that has a principal that is the UserDetails returned by the configured UserDetailsService.

11. Which of the following paths will be matched by mvcMatchers(/*/employees”)?

1. /management/employees

2. /hr/employees.html

3. /hr/employees

4. /hr/archived/employees

Correct Answer: 1, 2, 3

Explanation: “The last one is not matched because there are 2 URL parts (““hr”” and ““archived””) before employees, but only 1 is permitted in the pattern.

Since mvcMatchers was used, paths that would match but have a file extension (such as .html or .css), will also be matched (unlike antMatchers).

12. Which of the following are authentication mechanisms provided by Spring Security?

1. JMS Login

2. OAuth 2.0

3. Form Login

4. All of the above.

Correct Answer: 2, 3

Explanation: "Spring Security offers the following authentication mechanisms: Username and Password, OAuth 2.0, SAML 2.0, CAS, Remember Me, JAAS Authentication, OpenID, Pre-Authentication Scenarios and X509 Authentication.

Testing

1. In a spring test, which annotation can be used to mock a user?

1. @Mock

2. @MockBean

3. @WithMockUser

4. @MockAuthentication

Correct Answer: 3

Explanation: @WithMockUser is the most commonly used way of mocking a user in tests. Others possibilities are using the annotations @WithUserDetails, @WithAnonymousUser or @WithSecurityContext.

2. Using _____ makes it easier to mock classes in tests

1. Spring AOP

2. The template design pattern

3. Spring profiles

4. Java interfaces

Correct Answer: 4

Explanation: "Java interfaces are useful in tests, because they can easily be replaced with mocks or stubs in tests.

3. Which of the following can be used to execute SQL scripts before running test methods?

1. @Sql annotation

2. @SqlScript annotation

3. runScript method of the SqlRunner class

4. addScripts method of the ResourceDatabasePopulator class

Correct Answer: 1, 4

Explanation: "In addition to the aforementioned mechanisms for running SQL scripts programmatically, you can declaratively configure SQL scripts in the Spring TestContext Framework. Specifically, you can declare the @Sql annotation on a test class or test method to configure individual SQL statements or the resource paths to SQL scripts that should be run against a given database before or after an integration test method. Support for @Sql is provided by the SqlScriptsTestExecutionListener, which is enabled by default.

ResourceDatabasePopulator provides an object-based API for programmatically populating, initializing, or cleaning up a database by using SQL scripts defined in external resources. ResourceDatabasePopulator provides options for configuring the character encoding, statement separator, comment delimiters, and error handling flags used when parsing and running the scripts. Each of the configuration options has a reasonable default value. See the javadoc for details on default values.

4. Which of these annotation is used to annotate test classes and determines how the spring application context that will be available to all tests in the class is to be loaded and configured?

1. @Configuration

2. @ContextConfiguration

3. @ContextProperties

4. @Bean

Correct Answer: 2

Explanation: The @ContextConfiguration annotation can be used to configure the Application Context in Tests.

5. _____ testing refers to automated tests for several modules and the interactions between them when they are put together.

1. Unit

2. Integration

3. Performance

4. Interaction

Correct Answer: 2

Explanation: "Integration testing (sometimes called integration and testing, abbreviated I&T) is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing.

6. Which of the following are valid ways to initialize MockMvc in a Spring Test

1. mockMvc = MockMvcBuilders.
webAppContextSetup(this.wac).build();

2. mockMvc = new MockMvc();

3. mockMvc = MockMvc.create();

4. mockMvc = MockMvcBuilders.
standaloneSetup(new AccountController()).

build();

Correct Answer: 1,4

Explanation: "MockMvcBuilders contains factory methods for initializing MockMvc.

MockMvc does not have a public constructor.

create method in MockMvc does not exist.

7. Which of the following objects should be used in Spring tests for server-side testing?

1. TestRestTemplate

2. RestTemplate

3. MockMvc

4. MvcTester

Correct Answer: 3

Explanation: "TestRestTemplate is used for client-side testing (wherever RestTemplate is normally used in the code).

RestTemplate is not recommended for normal use in test classes. MvcTester does not exist.

MockMvc can be used to mock usage of HTTP endpoints and also has methods for checking the

result. It also features a fluent API.

Here is an example of MockMvc usage in a Test method:

```
this.mockMvc.perform(get(""/accounts/1"")

    .accept(MediaType.APPLICATION_JSON))

.andExpect(status().isOk())

.andExpect(content().
contentType(""application/json""))

.andExpect(jsonPath(""$.name"").
value(""Lee""));
```

8. Which of the following are true about Spring Framework's Test module?

1. AssertJ is included as a dependency

2. JUnit is supported

3. It supports Mockito Framework

4. It supports EasyMock

Correct Answer: 2, 3, 4

Explanation: "Spring Framework's Test module offers support for JUnit, Mockito and EasyMock.

AssertJ is an AOP dependency which is not

included or required for the Spring Test Module.

Transaction

1. Which is the most restrictive database system isolation level?

1. READ_UNCOMMITED

2. REPEATABLE_READ

3. SERIALIZABLE

4. READ_COMMITED

Correct Answer: 3

Explanation: "SERIALIZABLE isolation level is the most restrictive of all isolation levels. Transactions are executed with locking at all levels (read, range and write locking) so they appear as if they were executed in a serialized way. This leads to a scenario where none of the issues mentioned above may occur, but in the other way we don't allow transaction concurrency and consequently introduce a performance penalty.

2. Which is the annotation where transactions configuration properties such as isolation, propagation and rollbackFor can be configured?

1. @EnableTransaction

2. @Transaction

3. @TransactionConfiguration

4. @Transactional

Correct Answer: 4

Explanation: @Transactional is the main annotation used for declarative transactional management. It has all of these attributes.

3. A _____ is an operation that consists of number of tasks that take place as single unit- either all tasks are performed or no tasks are performed.

1. Data Validation

2. Transaction

3. HTTP Request

4. Logging

Correct Answer: 2

Explanation: "Transactions are a group of tasks that are either performed or aborted (in which case any changes will be rolled back).

4. Which of the following implementations of PlatformTransactionManager can be used with EclipseLink JPA?

1. JtaTransactionManager

2. JmsTransactionManager

3. HibernateTransactionManager

4. JpaTransactionManager

Correct Answer: 1, 4

Explanation: "JTA transaction manager can be used for JPA as well (since it can access multiple transactional resources)

5. Which of the following transaction APIs are supported by Spring?

1. JTA

2. JPA

3. JDO

4. Hibernate

Correct Answer: 1, 2, 3, 4

Explanation: Spring offers a consistent programming model across different transaction APIs, such as Java Transaction API (JTA), JDBC, Hibernate, and the Java Persistence API (JPA).

6. Using the default configurations, which methods will be affected when adding the

@Transactional annotation to the class?

1. public methods

2. protected methods

3. private methods

4. @Transactional may be used only on methods

Correct Answer: 1

Explanation: Since declarative transactional management is made possible using Spring AOP, which by default uses JDK Dynamic Proxies, which can only intercept public methods called from outside the class.

7. Which of the following transaction propagation behavior use the current transaction if there is one?

1. MANDATORY

2. REQUIRED

3. NEVER

4. SUPPORTS

Correct Answer: 1 2, 4

Explanation: All of these use the current transaction if there is one except for NEVER, which

never uses transactions.

8. _____ means that transactions are completely independent from one another?

1. Atomicity

2. Consistency

3. Isolation

4. Durability

Correct Answer: 3

Explanation: The effect of one transaction will not have any impact on another transaction so they are independent to one another. They are totally isolated from one another.

9. What are transactions taking place within the same resource (same database) called?

1. Internal Transactions

2. Local Transactions

3. Confined Transactions

4. Scoped Transactions

Correct Answer: 2

Explanation: Local transactions are resource-specific, such as a transaction associated with a JDBC connection.

10. On which of these database system isolation levels can non-repeatable reads happen?

1. REPEATABLE_READ

2. READ_UNCOMMITED

3. READ_COMMITED

4. SERIALIZABLE

Correct Answer: 2, 3

Explanation: Both READ_COMMITTED and READ_UNCOMMITTED allow for non-repeatable reads to happen.

11. Which of the following exceptions, when thrown, will cause a Spring transaction rollback by default?

1. RuntimeException

2. Checked exceptions

3. Error

4. None of the above.

Correct Answer: 1, 3

Explanation: Checked (business) exceptions will not cause transaction rollback.

12. Which of the following transaction propagation behavior will create a new transaction if one does not already exist?

1. MANDATORY

2. REQUIRED

3. NESTED

4. SUPPORTS

Correct Answer: 2, 3

Explanation: If there is no current transaction, NESTED acts like REQUIRED and creates a new transaction.

MANDATORY will throw an exception in this case, whereas SUPPORTS will execute the code non-transactionally.

13. Declarative Transaction Management in Spring Applications is implemented using _____?

1. Spring JDBC

2. Spring AOP

3. Spring Security

4. Spring Data Rest

Correct Answer: 2

Explanation: Transactions is a cross cutting concern which is implemented using AOP.

14. _____ means that either all changes or none of the changes in a transaction are applied?

1. Atomicity

2. Consistency

3. Isolation

4. Durability

Correct Answer: 1

Explanation: "Transaction follows ACID properties, where A stands for Atomicity. According to it, transactions should be either in Success state or Failure state.

15. What are transactions working with multiple transactional resources (relational databases or message queues) called?

1. Global transactions

2. Generic transactions

3. Event transactions

4. General transactions

Correct Answer: 1

Explanation: "Global transactions enable you to work with multiple transactional resources, typically relational databases and message queues.

Previously, the preferred way to use global transactions was via EJB CMT (Container Managed Transaction)

Full Length Practice Test

1. Which of the following Spring MVC-related information types are collected in metrics by Spring Boot Actuator by default?

1. Requesting user

2. HTTP method

3. Accessed endpoint

4. Response status

Correct Answer: 2,3,4

Explanation: By default, Spring MVC-related metrics are tagged with the following information:

- exception - Simple class name of any exception that was thrown while handling the request
- method - Request' s method (for example, GET or POST)
- outcome - Request's outcome based on the status code of the response. 1xx is INFORMATIONAL, 2xx is SUCCESS, 3xx is REDIRECTION, 4xx CLIENT_ERROR, and 5xx is SERVER_ERROR
- status - Response's HTTP status code (for example, 200 or 500)
- uri - Requests URI template prior to variable substitution, if possible (for example, /api/person/{id})

2. What do SpEL expressions starting with # reference?

1. Properties in the application environment

2. Spring Beans

3. Literal Values

4. JVM Properties

Correct Answer: 2

Explanation: "A Spring bean is referenced using its name prefixed with @ in SpEL. Chains of property references can be accessed using the period character.

- Example accessing property on Spring bean: @mySuperComponent.injectedValue

- Example invoking method on Spring bean: @mySuperComponent.toString()

3. Which of the following methods will be called first in the bean lifecycle?

1. afterPropertiesSet method in InitializingBean interface

2. init-method as specified in the Spring XML Configuration

3. Any methods annotated with @PostConstruct

4. Any method named "init"

Correct Answer: 3

Explanation: For each bean in the container the lifecycle happens as follows:

An instance of the bean is created using the bean metadata.

Properties and dependencies of the bean are set.

Any instances of BeanPostProcessor are given a chance to process the new bean instance before and after initialization.

- Any methods in the bean implementation class annotated with @PostConstruct are invoked.

This processing is performed by a BeanPostProcessor.

- Any afterPropertiesSet method in a bean implementation class implementing the

InitializingBean interface is invoked.

This processing is performed by a BeanPostProcessor. If the same initialization method has already been invoked, it will not be invoked again.

- Any custom bean initialization method is invoked.

Bean initialization methods can be specified either in the value of the init-method attribute in the corresponding <bean> element in a Spring XML configuration or in the initMethod property of the @Bean annotation.

This processing is performed by a BeanPostProcessor. If the same initialization method has already been invoked, it will not be

invoked again.

- The bean is ready for use.

4. Which of the following are auto-configured when using @DataJpaTest?

1. Spring Repositories

2. Spring Security

3. DataSource

4. Message source

Correct Answer: 1, 3

Explanation: "The @DataJpaTest annotation auto-configures the following:

- Caching

- Spring Data JPA repositories

- Flyway database migration tool

- A DataSource - The data-source will, as default, use an embedded in-memory database (test database).

- Data source transaction manager - A transaction manager for a single DataSource.

- A JdbcTemplate

- Liquibase database migration tool

- JPA based configuration for Hibernate

- Spring transaction

- A test database

- A JPA entity manager for tests

5. What one of these ensures that if anything goes wrong, the changes will be preserved once the system is back?

1. Atomicity

2. Consistency

3. Isolation

4. Durability

Correct Answer: 4

Explanation: The effect of one transaction will not have any impact on another transaction so they are independent to one another. They are totally isolated from one another.

6. Which of the following properties are required in order to configure an external MySQL Database?

1. spring.datasource.password

2. spring.datasource.username

3. spring.datasource.url

4. spring.datasource.driver-class-name

Correct Answer: 1,2,3,4

Explanation: All of these are required. Depending on the Database Provider, sometimes it is not necessary to provide the driver-class-name.

7. Which class is used for programmatic usage of transactions?

1. TransactionTemplate

2. TransactionExecutor

3. @Transactional

4. RollbackManager

Correct Answer: 1

Explanation: @Transactional annotation in spring is used for declarative usage, not programmatic.

8. What effect does setting the attribute "readOnly" on the @Transactional annotation to true have?

1. It does not allow write operations

2. It may optimize query performance

3. Sets the lock mode to READ

4. Nothing. It is there only for documentation.

Correct Answer: 2

Explanation: Only some databases reject the INSERT and UPDATE statements inside a read only transaction.

9. Which of the following does Spring Boot provide regarding error handling?

1. Global error page

2. JSON error response

3. Checked exceptions for most common problems

4. Logging errors stack traces separately

Correct Answer: 1, 2

Explanation: By default, Spring Boot provides and /error mapping that handles all errors in a sensible way, and it is registered as a "global error page" in the servlet container. For machine clients, it produces a JSON response with details of the error, the HTTP status, and the exception message. For browser clients, there is a "whitelabel" error view that renders the same data in HTML format

(to customize it, add a View that resolves to error). Spring only throws unchecked exceptions in the case of problems.

10. Which of the following are web environment options offered by @ SpringBootTest?

1. RANDOM_PORT

2. WEB

3. MINIMAL

4. DEFINED_PORT

Correct Answer: 1, 4

Explanation: Four different types of web environments can be specified using the webEnvironment attribute of the @SpringBootTest annotation:

- MOCK - Loads a web ApplicationContext and provides a mock web environment. Does not start a web server.

- RANDOM_PORT - Loads a WebServerApplicationContext, provides a real web environment and starts an embedded web server listening on a random port. The port allocated can be obtained using the @ LocalServerPort annotation or @Value(""${local.server.port}""). Web server runs in a separate

thread and server-side transactions will not be rolled back in transactional tests.

- DEFINED_PORT - Loads a WebServerApplicationContext, provides a real web environment and starts an embedded web server listening on the port configured in the application properties, or port 8080 if no such configuration exists. Web server runs in a separate thread and server-side transactions will not be rolled back in transactional tests

- NONE - Loads an ApplicationContext without providing any web environment.

11. On which of these database system isolation levels can phantom reads occur?

1. READ_UNCOMMITED

2. SERIALIZABLE

3. REPEATABLE_READ

4. READ_COMMITED

Correct Answer: 1, 3, 4

Explanation: Phantom reads can occur in all database system isolation levels except for SERIALIZABLE.

12. Which of the following are valid ways

of adding a Bean definition to the IoC Container?

1. <bean/> in XML

2. Calling DefaultListableBeanFactory. registerBeanDefinition

3. @Bean annotated method in a @Configuration class

4. Java Class annotated with @Component

Correct Answer: 1,2,3,4

Explanation: All of these are valid ways of adding a bean definition to the IoC Container.

13. In order to define a bean, one can create a class annotated with _____ and add a method annotated with _____ to it.

1. @Service @BeanDefinition

2. @Configuration, @Bean

3. @Configuration, @Inject

4. @Component, @ManagedBean

Correct Answer: 2

Explanation: The Annotation @BeanDefinition does not exist. The annotation @ManagedBean is not part of Spring (it is a JEE class annotation similar to @Component).

14. Which of the following are true about Spring MVC Controllers?

1. When using annotations, specific class or interface inheritance is not required.

2. Endpoint paths mapped by a controller must all have the same prefix.

3. @Controller is a stereotype annotation.

4. Controllers implemented using annotations do not have direct dependencies on Portlet or Servlet APIs.

Correct Answer: 1,3,4

Explanation: Endpoint Paths do not necessarily need to have the same path prefix.

@Controller is a stereotype annotation because it is Meta annotated with @Component.

Before annotation based controller declaration, controllers needed to extend certain classes, such as MultiActionController.

15. What does the abbreviation MVC stand for?

1. Multi Variable Control

2. Multi Variable Compiler

3. Model View Compiler

4. Model-View-Controller

Correct Answer: 4

Explanation: MVC stands for Model-View-Controller. It is a programming paradigm which provides separation of concern, which makes it easy to develop and maintain a web application.

16. Which of the following configuration properties must be changed to allow bean definition overriding?

1. spring.beaninfo.ignore

2. management.endpoint.beans.enabled

3. spring.main.allow-bean-definition-overriding

4. Bean Definition Overriding is allowed by default

Correct Answer: 3

Explanation: Spring bean definition overriding is not on by default. The first answer refers to BeanInfo configuration, the second to default actuator properties.

17. Which of the following are condition annotations used by Spring Boot for auto-configuration?

1. @ConditionalOnResource

2. @ConditionalOnProperty

3. @ConditionalOnMissingBean

4. @ConditionalOnCloudPlatform

Correct Answer:1,2,3,4

Explanation: Here are all of the possible auto-configuration condition annotations:

- @ConditionalOnClass - Presence of class on classpath.

- @ConditionalOnMissingClass - Absence of class on classpath.

- @ConditionalOnBean - Presence of Spring bean or bean type (class).

- @ConditionalOnMissingBean - Absence of Spring bean or bean type (class).

- @ConditionalOnProperty - Presence of Spring environment property.

- @ConditionalOnResource - Presence of

resource such as file.

- @ConditionalOnWebApplication - If the application is considered to be a web application, that is uses the Spring WebApplicationContext, defines a session scope or has a StandardServletEnvironment.

- @ConditionalOnNotWebApplication - If the application is not considered to be a web application.

- @ConditionalOnExpression - Bean or configuration active based on the evaluation of a SpEL expression.

- @ConditionalOnCloudPlatform - If specified cloud platform, Cloud Foundry, Heroku or SAP, is active.

- @ConditionalOnEnabledEndpoint - Specified endpoint is enabled.

- @ConditionalOnEnabledHealthIndicator - Named health indicator is enabled.

- @ConditionalOnEnabledInfoContributor - Named info contributor is enabled.

- @ConditionalOnEnabledResourceChain - Spring resource handling chain is enabled.

- @ConditionalOnInitializedRestarter - Spring DevTools RestartInitializer has been applied with

non-null URLs.

- @ConditionalOnJava - Presence of a JVM of a certain version or within Condition Annotation Condition Factor a version range.

- @ConditionalOnJndi - Availability of JNDI InitialContext and specified JNDI locations exist.

- @ConditionalOnManagementPort - Spring Boot Actuator management port is either: Different from server port, same as server port or disabled.

- @ConditionalOnRepositoryType - Specified type of Spring Data repository has been enabled.

- @ConditionalOnSingleCandidate - Spring bean of specified type (class) contained in bean factory and single candidate can be determined.

18. Which of these protocols is used to access Spring Boot actuator endpoints?

1. SOAP

2. JMX

3. FTP

4. HTTP

Correct Answer: 2, 4

Explanation: Both JMX and HTTP protocol can be used to access Spring Boot Actuator endpoints. You can choose to manage and monitor your application by using HTTP endpoints or with JMX. Auditing, health, and metrics gathering can also be automatically applied to your application.

19. _____ is linking aspects with other application types or objects to create an advised object?

1. Advice

2. Aspect

3. Weaving

4. Pointcut

Correct Answer: 1

Explanation: "Weaving: linking aspects with other application types or objects to create an advised object. This can be done at compile time (using the AspectJ compiler, for example), load time, or at runtime. Spring AOP, like other pure Java AOP frameworks, performs weaving at runtime.

20. Which exception can potentially be thrown when calling the JdbcTemplate. query() method in Spring?

1. SQLException

2. QueryException

3. JdbcException

4. DataAccessException

Correct Answer: 4

Explanation: DataAccessException is the root exception thrown by all Data Access methods. Any SQLExceptions will be suppressed by a DataAccessException.

21. Which of the following is a feature of Spring Boot Actuator?

1. Monitoring

2. Metrics

3. Management

4. Circuit-Breaker

Correct Answer: 1, 2, 3

Explanation: Monitoring, Metrics and Management are provided by Spring Boot Actuator. Ribbon and Hystrix are the tools provided by Spring as circuit breakers.

22. Which of the following symbols is used in @Value expressions?

1. &

2. $

3. ^

4. #

Correct Answer: 2, 4

Explanation: "Expressions starting with $.

Such expressions reference a property name in the applications environment. These expressions are evaluated by the PropertySourcesPlaceholderConfigurer Spring bean prior to bean creation and can only be used in @Value annotations.

On the other hand, Expressions starting with # are Spring Expression Language expressions parsed by a SpEL expression parser and evaluated by a SpEL expression instance.

23. In Spring Security, how can you enable the @Secured annotations ?

1. @EnableGlobalMethodSecurity(prePostEnabled = true)

2. @EnableGlobalMethodSecurity(securedEnabled = true)

3. @EnableGlobalMethodSecurity(jsr250Enabled = true)

4. They are enabled by default.

Correct Answer: 2

Explanation: "The attribute securedEnabled specifies whether the @Secured annotation can be used in the application. You can read more about how to enable Spring Security in my earlier post about the same.

24. Which of the following HTTP method is idempotent?

1. GET

2. PUT

3. POST

4. DELETE

Correct Answer: 1, 2, 4

Explanation: The GET, PUT and DELETE methods are idempotent, meaning that applying them multiple times to a resource results in the same state change of the resource as applying them once, though the response might differ.

25. Which of the following methods in JdbcTemplate is recommended for UPDATE or INSERT operations?

1. query

2. update

3. execute

4. queryForList

Correct Answer: 2

Explanation: update should be used for write operations while query should be for select queries. You can read more about JdbcTemplate in my article 10 examples of JdbcTempalte and see the examples of all these methods to learn more.

26. Which Interface which is responsible for Instantiating, Configuring, Assembling and Managing the life-cycle of spring beans."

1. ApplicationContext

2. ApplicationBeanContext

3. ApplicationFactoryContext

4. BeanContext

Correct Answer: 1

Explanation: ApplicationContext is an Interface, which acts as a container in spring and is responsible for carrying out the life cycle of beans. You can read more about that in my post difference between BeanRepository and ApplicationContext

27. Spring has mock objects to use in tests in the following areas?

1. RMI

2. Environment

3. JMX

4. JNDI

Correct Answer: 2, 4

Explanation: Spring has mock objects on Environment, JNDI, and Servlet API to assist in unit testing.

28. Consider the following code of a bean definition method:

@Bean

public BeanX beanY(BeanZ beanZ)

What is the name of the created bean?"

1. BeanX

2. beanX

3. beanY

4. beanZ

Correct Answer: 3

Explanation: When defined this way, the name of the method will same as the bean name. You can also read my tutorial on @Bean annotation to learn more about it.

29. Which class associates a request URL pattern with a list of filters in Spring Security?

1. FilterChainProxy

2. FilterChain

3. DelegatingFilterProxy

4. SecurityFilterChain

Correct Answer: 4

Explanation: SecurityFilterChain is used by FilterChainProxy to determine which Spring Security Filters should be invoked for this request.

30. Which options of dependency injection exist?

1. constructor-based

2. template-based

3. field-based

4. setter-based

Correct Answer: 1, 3, 4

Explanation: Template-based injection does not exist

31. Which of the following AOP advice types can choose to prevent execution of the advised method, and instead return another value?

1. @Before

2. @Around

3. @AfterThrowing

4. @AfterReturning

Correct Answer: 2

Explanation: Before advice can also prevent execution of the advised method, however only if an exception is thrown. Around advice can prevent the execution of the advised method and change the return value.

32. What are the limitations of using the default JDK dynamic proxies in Spring AOP?

1. Method calls inside the same class cannot be intercepted.

2. There are fewer available pointcut designators.

3. Only public interface method calls can be intercepted

4. It cannot intercept constructor calls

Correct Answer: 3

33. Which of the following dependencies are included in the spring-boot-starter-test starter?

1. Hamcrest

2. EasyMock

3. JUnit

4. PowerMock

Correct Answer: 1, 3

Explanation: The spring-boot-starter-test starter adds the following test-scoped dependencies to the classpath:

- JUnit - Unit-testing framework.

- Spring Test and Spring Boot Test

- AssertJ - Fluent assertions for Java.

- Hamcrest - Framework for writing matchers that are both powerful and easy to read.

- Mockito - Mocking framework for Java.

- JSONassert - Tools for verifying JSON representation of data.

- JsonPath - A Java DSL for reading JSON documents.

34. Which of the following HTTP verbs can be used to create a new entity, but not modify an existing one?

1. GET

2. PUT

3. POST

4. DELETE

Correct Answer: 3

Explanation: Function of POST: Create a member resource in the collection resource using the instructions in the request body. The URI of the created member resource is automatically assigned and returned in the response Location header field.

35. Which of the following are valid ways to inject a dependency into a bean in Spring?

1. BeanFactory.getBean() method

2. @Autowired annotation

3. XML Configuration

4. Constructor argument

Correct Answer: 1,2,3,4

Explanation: All of these are valid ways.

36. "Consider the code sample below:

```
public class MyClass {
       @MyAnnotation
       public String myMethod(){
        ///
      }
```

Fill in the blank in the pointcut expression below, so that it matches the above method:

_____(MyAnnotation)

1. @target

2. @args

3. @within

4. @annotation

Correct Answer: 4

Explanation: @target limits matching to join points (the execution of methods when using Spring AOP) where the class of the executing object has an annotation of the given type. While @args limits matching to join points (the execution of methods when using Spring AOP)

37. What is the name of the default prefix of all actuator endpoints?

1. endpoint/actuator/

2. management/

3. actuator/metrics/

4. actuator/

Correct Answer: 4

Explanation: "The default prefix for actuator endpoints is ""actuator/"

38. Which of these is the correct naming convention for custom find methods in Spring Data Repository Interface?

1. find(First[count]) [ordering operator] [comparison operator] By[Property Expression]

2. find(First[count]) By[Property Expression] [ordering operator] [comparison operator]

3. find(First[count]) By[Property Expression] [comparison operator] [ordering operator]

4. None of the above.

Correct Answer: 3

Explanation: This is a tricky Spring Data JPA question but you need to remember that Result limit (First/Last X) comes first. After By comes to the property to be queried, followed by the comparison operator. The sort order comes last.

39. Which of these act as Spring Boot dependency descriptors?

1. Starters

2. Actuator

3. Dependencies

4. Reactor

Correct Answer: 1

Explanation: "Starters are a set of convenient dependency descriptors that you can include in your application. You get a one-stop-shop for all the Spring and related technologies that you need without having to hunt through sample code and copy-paste loads of dependency descriptors. For example, if you want to get started using Spring and JPA for database access, include the spring-

boot-starter-data-jpa dependency in your project.

40. A HTTP Response Code of _____ means _____

1. 500, Server Side Error

2. 400, Client Side Error

3. 300, Success

3. 200, Client Side Error

Correct Answer: 1, 2

Explanation: Http Response Status Codes for a request are 1x-informational, 2x- Success, 3x-Redirectional, 4x-Client Side Error, 5x-Server Side Error.

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

1. Method parameters can be annotated with @Mock

2. @Mock is included without additional dependencies

3. @Mock can be used for non-bean classes

4. @Mock can be used without SpringRunner

Correct Answer: 1, 3, 4

Explanation: "Both the @MockBean and @Mock annotation can be used to create Mockito mocks but there are a couple of differences between the two annotations:

- @Mock can only be applied to fields and parameters while @MockBean can only be applied to classes and fields.

- @Mock can be used to mock any Java class or interface while @MockBean only allows for mocking of Spring beans or creation of mock Spring beans.

- @MockBean can be used to mock existing beans but also to create new beans that will belong to the Spring application context.

- To be able to use the @MockBean annotation, the Spring runner (@RunWith(SpringRunner. class)) has to be used to run the associated test.

- @MockBean can be used to create custom annotations for specific, reoccurring, needs when running Spring Boot Tests, both @Mock and @MockBean are included in the spring-boot-starter-test.

42. Which technology is used to accomplish method-level security in Spring?

1. Transactions

2. Servlet Filters

3. AOP

4. None of the above.

Correct Answer: 3

Explanation: Method security is based on Spring AOP

43. Which of the following does Spring HATEOAS auto-configuration provide?

1. HateoasController

2. ObjectMapper

3. HATEOAS Security

4. @EnableHypermediaSupport

Correct Answer: 2, 4

Explanation: If you develop a RESTful API that makes use of hypermedia, Spring Boot provides auto-configuration for Spring HATEOAS that works well with most applications. The auto-configuration replaces the need to use @ EnableHypermediaSupport and registers a number of beans to ease building hypermedia-based applications, including a LinkDiscoverers

(for client-side support) and an ObjectMapper configured to correctly marshal responses into the desired representation.

The ObjectMapper is customized by setting the various spring.jackson.* properties or, if one exists, by a Jackson2ObjectMapperBuilder bean.

You can take control of Spring HATEOAS' configuration by using @EnableHypermediaSupport. Note that doing so disables the ObjectMapper customization described earlier.

44. Which of these Annotations enables Spring Boot auto-configuration?

1. @Configuration

2. @EnableAutoConfiguration

3. @AutoConfiguration

4. @SpringConfiguration

Correct Answer: 2

Explanation: The @EnableAutoConfiguration is used to enable Spring Boot's auto-configuration mechanism. While @Configuration is an annotation used on classes which contain bean definition methods. You can also use @SpringBootApplication to enable auto-configuration in Spring Boot applications.

45. Which of the following is provided by Spring Boot's Spring MVC Auto-configuration?

1. ContentNegotiatingViewResolver bean

2. RestController testing pages

3. HttpMessageConverters support

4. Support for serving static resources

Correct Answer: 1, 3, 4

Explanation: "Spring Boot provides auto-configuration for Spring MVC that works well with most applications.

The auto-configuration adds the following features on top of Spring's defaults:

- Inclusion of ContentNegotiatingViewResolver and BeanNameViewResolver beans.

- Support for serving static resources, including support for WebJars (covered later in this document)).

- Automatic registration of Converter, GenericConverter, and Formatter beans.

- Support for HttpMessageConverters (covered later in this document).

- Automatic registration of MessageCodesResolver (covered later in this document).

- Static index.html support.

- Custom Favicon support (covered later in this document).

- Automatic use of a ConfigurableWebBindingInitializer bean (covered later in this document).

46. Which of the following HTTP methods can be mapped to a method annotated with @RequestMapping?

1. GET

2. PUT

3. POST

4. DELETE

Correct Answer: 1,2,3,4

Explanation: "RequestMapping is an annotation for mapping web requests onto methods in request-handling classes with flexible method signatures.

By default, it will map all HTTP methods. One can specify which methods should be allowed using

the “method” attribute or use one of the HTTP method variants such as @GetMapping.

47. Which of the following bean scopes can only be used in the context of a web-aware Spring Application Context?

1. web

2. request

3. session

4. application

Correct Answer: 2, 3, 4

Explanation: “The following bean scopes are only valid in the context of a web-aware Spring ApplicationContext: session, request, application, WebSocket. There is no ‘web’ bean scope out of the box.

48. Which of the following annotations can be used to inject property values into Spring beans and configuration classes?

1. @Primary

2. @Value

3. @Import

4. @PropertyValue

Correct Answer: 2

Explanation: The @Value is an annotation used to inject property values into a field using SpEL.

49. What happens if you define both the "id" and "name" attributes in a bean's XML definition?

1. "id" is used as a bean identifier

2. "name" is used for bean aliases

3. An exception is thrown

4. The application will crash.

Correct Answer: 1,2

Explanation: XML-based configuration metadata, you use the id attribute, the name attribute, or both to specify the bean identifiers. The id attribute lets you specify exactly one id. Conventionally, these names are alphanumeric ('myBean', 'someService', etc.), but they can contain special characters as well. If you want to introduce other aliases for the bean, you can also specify them in the name attribute, separated by a comma (,), semicolon (;), or white space.

50. Which of these testing refers to automated tests for the smallest unit of functionality. This may be a method in a class, a class or even an entire module

1. Unit

2. Integration

3. Performance

4. Interaction

Correct Answer: 1

Explanation: Unit testing is the foremost testing done in the software development life cycle. All individual units of an application are tested to make sure that they are working well.

That's all about the Spring Certification Practice Questions. You can use this practice test to check your knowledge as well as learn essential spring concepts and topics for spring professional certifications. All the best for your Spring Professional Certification

Index

Symbols

A

B

C

D

E

H

J

M

P

R

S

T

U