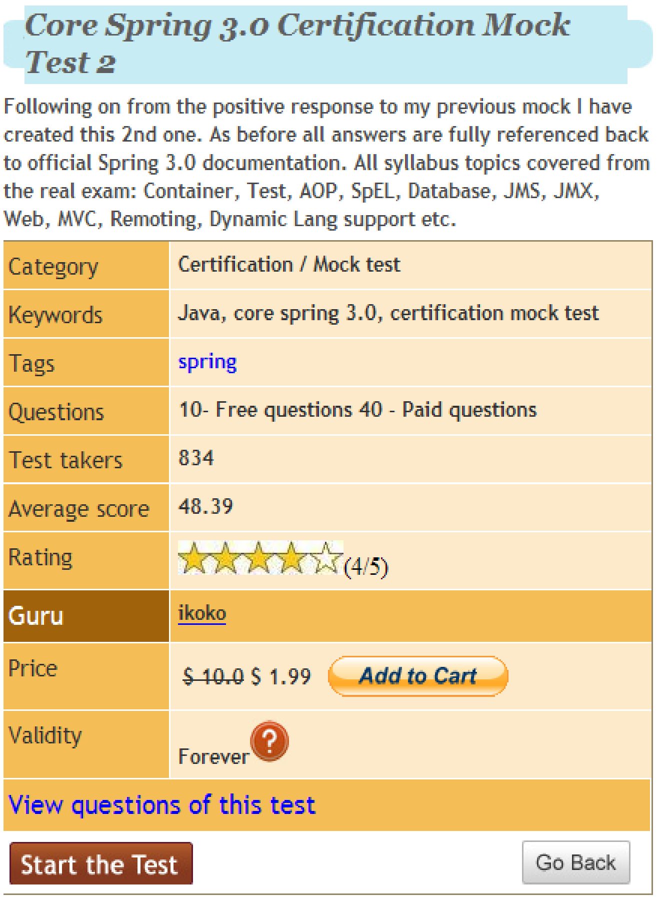
http://www.skill-guru.com/test/86/core-spring-3.0-certification-mock-test-2



Core Spring 3.0 Certification Mock Test 2

Following on from the positive response to my previous mock I have created this 2nd one. As before all answers are fully referenced back to official Spring 3.0 documentation. All syllabus topics covered from the real exam: Container, Test, AOP, SpEL, Database, JMS, JMX, Web, MVC, Remoting, Dynamic Lang support etc.

Category	Certification / Mock test
Keywords	Java, core spring 3.0, certification mock test
Tags	spring
Questions	10- Free questions 40 - Paid questions
Test takers	834
Average score	48.39
Rating	★★★★☆ (4/5)
Guru	ikoko
Price	~~$ 10.0~~ $ 1.99 Add to Cart
Validity	Forever ?

View questions of this test

Start the Test   Go Back

Test : Core Spring 3.0 Certification Mock Test 2

Question :1

Web MVC

Which of the following two options would be best practises if you require your controller to issue a redirect to the browser?

Correct Answer	Your Selection	Answer	Explanation
	☐	Just simply utilise the HttpServletResponse.sendRedirect() call.	INCORRECT: This would work but breaks the MVC pattern and is not a best practise.
✓	☐	Create and return an instance of Spring's RedirectView.	CORRECT: From the reference manual ... "The RedirectView issues an HttpServletResponse.sendRedirect() call that returns to the client browser as an HTTP redirect. All model attributes are exposed as HTTP query parameters. This means that the model must contain only objects (generally Strings or objects converted to a String representation), which can be readily converted to a textual HTTP query parameter." Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/mvc.html#mvc-redirecting-redirect-view
✓	☐	Return a view name that is prefixed with "redirect:"	CORRECT: In fact this is the most preferred of all because the Controller is unaware that it is actually resulting in a redirect (assuming the view name prefixed with "redirect:" is injected. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/mvc.html#mvc-redirecting-redirect-prefix
	☐	Simply call super.redirect() on the Spring BaseController that you extend	INCORRECT: This is wrong for several reasons. Your controller does not need to extend any Spring base controller. There is no class called BaseController. Finally, therefore it follows there is no option to call super.redirect().

Question :2 AOP

If some target code (that is matched to a pointcut expression) throws an exception - which of the following advices would NOT execute?

Correct Answer	Your Selection	Answer	Explanation
	○	An advice using the @After annotation (with a matching pointcut expression)	INCORRECT: This advice would execute as it equates to after (finally) so would run in both normal and exception scenarios. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/aop.html#aop-schema-advice-after-finally
	○	An advice using the @Before annotation (with a matching pointcut expression)	INCORRECT: This advice would execute as it runs before its target executes. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/aop.html#aop-advice-before
✓	○	An advice using the @AfterReturning annotation (with a matching pointcut expression)	CORRECT: This advice would NOT execute as it would only run in a normal scenario with no exceptions. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/aop.html#aop-schema-advice-after-returning
	○	An advice using the @AfterThrowing annotation (with a matching pointcut expression)	INCORRECT: This advice would execute as it would run after an exception was thrown from the target. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/aop.html#aop-schema-advice-after-throwing

Question :3 AOP

Load-time weaving (LTW) is the process of weaving AspectJ aspects into an application's class files as they are being loaded into the Java virtual machine (JVM).

Which of the following are *MANDATORY* steps in integrating AspectJ LTW in to your application?

1. Add the jars: `spring-aop.jar`, `aspectjrt.jar` and `aspectjweaver.jar` in to your classpath.
2. Create you Aspect class/es.
3. Create a `META-INF/aop.xml` that declares your aspect class/es.
4. Add `<context:load-time-weaver/>` in to your application's XML.

Correct Answer	Your Selection	Answer	Explanation
	○	1, 2, 3	INCORRECT: All four are mandatory.
	○	1, 3, 4	INCORRECT: All four are mandatory.
	○	2, 3, 4	INCORRECT: All four are mandatory.
✓	○	1, 2, 3, 4	CORRECT: 1. At a minimum you will need those 3 jars. If you are using the Spring-provided agent to enable instrumentation, you will also need spring-instrument.jar. 2. You at least must have one aspect and aspects are implemented as Java classes. 3. The AspectJ LTW infrastructure is configured using one or more 'META-INF/aop.xml' files. 4. This config line "switches on" the load-time weaving

Question :4 Database

Which of the following statements is TRUE regarding batch processing?

Correct Answer	Your Selection	Answer	Explanation
✓	○	Both JdbcTemplate and SimpleJdbcTemplate offer this feature via their batchUpdate() methods.	CORRECT: These 2 APIs may be used to achieve batch processing in Spring. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jdbc.html#jdbc-advanced-classic Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jdbc.html#jdbc-advanced-simple
	○	Spring advice to break out in to regular JDBC when batch processing as this feature is not supported in Spring.	INCORRECT: There is support for batch processing using the 2 key classes secpficied in the correct answer.
	○	Only JdbcTemplate offers this feature via its batchUpdate() method.	INCORRECT: SimpleJdbcTemplate also has this method. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jdbc.html#jdbc-advanced-simple
	○	You can achieve JdbcTemplate batch processing by implementing two methods of the interface, BatchPreparedStatementCreator, and passing this in as a second parameter in the batchUpdate method call.	INCORRECT: The interface that you need to implement is called BatchPreparedStatementSetter - otherwise this statement is true. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jdbc.html#jdbc-advanced-classic

Question :5 Container & Test

Which of the following statements regarding container related annotations are *TRUE*?

Correct Answer	Your Selection	Answer	Explanation
	○	@Inject has been largely superseded by the Java language @Autowired annotation	INCORRECT: @Inject is in fact part of the Java language. @Autowired is a Spring annotation. Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/beans/factory/annotation/Autowired.html
	○	@Autowired marks a method (typically a JavaBean setter method) as being 'required': that is, the setter method must be configured to be dependency-injected with a value.	INCORRECT: This describes the @Required annotation. Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/beans/factory/annotation/Required.html
	○	As of Spring 3.0 it is no longer necessary to define a BeanPostProcessor in your XML configuration to "switch on" annotations such as @Required and @Autowired.	INCORRECT: This is false. You still need to do this using classes such as RequiredAnnotationBeanPostProcessor and AutowiredAnnotationBeanPostProcessor to "switch on" your annotations in your source code. Another approach is to use the <context:annotation-config/> tag from the context namespace. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-annotation-config
✓	○	None of the above	CORRECT: None of the above are true - each answer gives an explanation above.

Question :6 Database

Spring's exception translation mechanism provides a consistent and runtime exception hierarchy to the client, abstracted away from the underlying data-access technology.

Correct Answer	Your Selection	Answer	Explanation
✓	○	TRUE	CORRECT: Spring provides a RuntimeException hierarchy to allow the client to choose whether to deal with irreversible exception scenarios - what's more this is a common hierarchy (DataAccessException) that hides the underlying technology specific exceptions. This includes JDBC as well as ORM technologies such as Hibernate. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/orm.html#orm-introduction
	○	FALSE	INCORRECT:

Question :7 SpEL

Which of the following 2 statements regarding SpEL are *true*?

Correct Answer	Your Selection	Answer	Explanation
✓	☐	The use of @Value("#{ systemProperties ['user.region'] }") will correctly locate the user.region from the system properties.	CORRECT: This will correctly access this property. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/expressions.html#expressions-beandef-annotation-based
	☐	The use of @Value("#{ system ['user.region'] }") will correctly locate the user.region from the system properties.	INCORRECT: The correct version is... @Value("#{ systemProperties ['user.region'] }") Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/expressions.html#expressions-beandef-annotation-based
	☐	The @Value annotation (with an embedded SpEL expression) may only be set on fields.	INCORRECT: The final statment explains this answer. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/expressions.html#expressions-beandef-annotation-based
✓	☐	The @Value annotation (with an embedded SpEL expression) may be set on to a field, setter method and on a parameter in a method that is annotated with @Autowired	CORRECT: This is a valid statement regarding the legal location of this annotation. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/expressions.html#expressions-beandef-annotation-based

Question :8 Container & Test

Lifecycle callbacks enable developers to execute code at certain points in a bean's lifecycle. However, there is a small downside in that you can only implement callbacks by coupling your application code to Spring.

Is this statement TRUE or FALSE?

Correct Answer	Your Selection	Answer	Explanation
	○	TRUE	INCORRECT
✓	○	FALSE	CORRECT: Although one approach to lifecycle callbacks requires your bean/s to implement InitializingBean and/or DisposableBean other approaches will NOT couple your code to Spring. These include configuring an "init-method" or a "destroy-method" in your configuration XML. Also Spring 3.0 supports the JSR-250 annotations: @PostConstruct and @PreDestroy which do exactlly the same. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-factory-lifecycle Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-postconstruct-and-predestroy-annotations

Question :9 **Remoting**

Spring 3.0 provides support for RESTful services.

Which of the following statements regarding these features of the framework are TRUE?

1. A `RestTemplate` is provided for accessing a RESTful service, which is analogous to a `JdbcTemplate`, `JmsTemplate` etc.
2. A RestServiceExporter is provided for exporting a RESTful service, which is analogous to a `HessianServiceExporter`, `RmiServiceExporter` etc.
3. `RestTemplate` provides support for ALL of the http methods: i.e. GET and POST as well as DELETE, HEAD, OPTIONS and PUT.
4. `RestTemplate` provides support for 2 of the 6 http methods: GET and POST only. You need to use a more tailored API, such as Jakarta Commons `HttpClient`, to gain full REST features.

Correct Answer	Your Selection	Answer	Explanation
	○	1, 2, 3	INCORRECT:
	○	1, 2, 4	INCORRECT:
✓	○	1, 3	CORRECT: 1. A RestTemplate provides you with a much less verbose way of dealing with a RESTful service, compared to an API such as Jakarta Commons HttpClient. 2. There is no RestServiceExporter and the framework reference does not cover how you provide a REST service in Spring. 3. Full support for all http methods is provided. 4. Is incorrect because 3 is correct.
	○	2, 3	INCORRECT:

Question :10 Transactions

Identify the correct statement below regarding annotation driven transactions.

Correct Answer	Your Selection	Answer	Explanation
	○	The presence of the @Transactional annotation is enough to activate the transactional behavior.	INCORRECT: The additional config line ... <tx:annotation-driven/> ... needs to be added to your XML. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/transaction.html#transaction-declarative-annotations
	○	Spring recommends that you annotate interfaces with the @Transactional annotation, as opposed to annotating concrete classes (and methods of concrete classes).	INCORRECT: Spring recommend the opposite. Java annotations are not inherited from interfaces. This means that your methods will not operate transactionally*. This is true if they are run as class-based proxies or using the weaving-based aspect. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/transaction.html#transaction-declarative-annotations
	○	In the config tag ... <tx:annotation-driven/> ... the optional "mode" attribute defaults to "aspectj". This weaves the classes with Spring's AspectJ transaction aspect, modifying the target class byte code to apply to any kind of method call.	INCORRECT: The default mode is "proxy". This processes annotated beans to be proxied using Spring's AOP framework. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/transaction.html#transaction-declarative-annotations
✓	○	In the config tag ... <tx:annotation-driven/> ... the "proxy-target-class" attribute allows two types of proxy mode and only applies when the "mode" is in "proxy" mode and not "aspectj".	CORRECT: This attribute may be true or false. True means class-based proxies are created. False means standard JDK interface-based proxies are created. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/transaction.html#transaction-declarative-annotations

Question :11 Container & Test

The `@Configuration` annotation is a new Java-centric way of configuring your Spring application. Which of the following statements regarding this annotation is *FALSE*?

Correct Answer	Your Selection	Answer	Explanation
✓	○	Spring's @Configuration class support is intended to be a 100% complete replacement for Spring XML.	CORRECT: This statement is FALSE. Spring's official documentation claims that this is not the case as there are still many useful features available in the various supported namespaces that may be leveraged in traditional XML configuration. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-java-combining
	○	You can "switch on" @Configuration annotations in your XML configuration with <context:annotation-config/>	INCORRECT: The question required you to specify the FALSE statement. This statement is TRUE.
	○	Because @Configuration is meta-annotated with @Component, annotated classes are automatically candidates for component scanning.	INCORRECT: The question required you to specify the FALSE statement. This statement is TRUE.
	○	@Configuration must be applied to a class only and not methods or fields.	INCORRECT: The question required you to specify the FALSE statement. This statement is TRUE.

Question :12 Web MVC -

In Spring's form tag library which of the following tags are valid?

1. input and inputs
2. checkbox and checkboxes
3. radiobutton and radiobuttons
4. redirect

Correct Answer	Your Selection	Answer	Explanation
	○	1, 2 & 3 only	
✓	○	2 & 3 only	CORRECT: There is an "input" tag, that renders an HTML 'input' tag with type 'text', but not an "inputs" tag. "checkbox" renders an HTML 'input' tag with type 'checkbox'. "checkboxes" renders multiple HTML 'input' tags with type 'checkbox'. "radiobutton" renders an HTML 'input' tag with type 'radio'. "radiobuttons" renders multiple HTML 'input' tags with type 'radio'. "redirect" does not exist and is not associated with rendering form structure anyway.
	○	1 & 4 only	
	○	All of the above	

Question :13 Resources - Which of the following statements regarding accessing low level resources using Spring are TRUE?

Correct Answer	Your Selection	Answer	Explanation
✓	☐	This API provides support for accessing files held on an FTP server.	CORRECT: UrlResource provides this ability to access files in the local filesystem or remotely over ftp and http. Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/core/io/UrlResource.html
	☐	Spring builds on the java.net.URL API and its ability to access files held on a classpath, and to verify whether a resource exists or not.	INCORRECT: Spring's Resource API provides these useful features that are lacking in the native Java API. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/resources.html#resources-introduction
	☐	Spring's Resource interface contains the following methods also on java.io.File, i.e. • boolean exists() • boolean delete() • File getAbsoluteFile() • boolean createNewFile()	INCORRECT: Spring's Resource API only has the first of these methods. The full list of methods are as follows: • boolean exists(); • boolean isOpen(); • URL getURL() throws IOException; • File getFile() throws IOException; • Resource createRelative(String relativePath) throws IOException; • String getFilename(); • String getDescription(); Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/resources.html#resources-resource Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/core/io/Resource.html
✓	☐	The Resource API is used extensively in Spring and wraps existing functionality where possible. For example, a UrlResource wraps a URL which is where the underlying work is done.	CORRECT: These statements are true. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/resources.html#resources-resource

Question :14 XML

Spring's OXM (Object XML mapping) APIs provide support for which of the following technologies?

1. `JAXB`

2. `Castor`

3. `XMLBeans`

4. `JiBX`

Correct Answer	Your Selection	Answer	Explanation
	○	None of the above	INCORRECT: See final answer option
	○	1 only	INCORRECT: See final answer option
	○	1 & 2 only	INCORRECT: See final answer option
	○	1, 2 & 3 only	INCORRECT: See final answer option
✓	○	All of the above	CORRECT: Spring abstracts all of these OXM implementors behind two interfaces Marshaller (to XML) and Unmarshaller (from XML) that define the following methods respectively: void marshal (Object graph, Result result) throws XmlMappingException, IOException; Object unmarshal(Source source) throws XmlMappingException, IOException; Spring provides the following classes in the org.springframework.oxm.xmlbeans package that in each case implement the Marshaller and Unmarshaller interfaces: Jaxb2Marshaller, CastorMarshaller, XmlBeansMarshaller, JibxMarshaller There is also similar support for a fifth OXM technology called XStream, via the XStreamMarshaller. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/oxm.html

Question :15 Database

The framework reference documentation recommends that `JdbcTemplate` is scoped as a "prototype" bean. This prevents thread concurrency issues when reading / writing to the database.

Correct Answer	Your Selection	Answer	Explanation
	○	TRUE	INCORRECT
✓	○	FALSE	CORRECT: It is perfectly fine to share one instance of JdbcTemplate with multiple DAOs. This bean does not maintain conversational state - except for its datasource property. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jdbc.html#jdbc-JdbcTemplate-idioms

Question :16 **Database**

The following methods ...

- `queryForInt(args)`
- `queryForList(args)`
- `getJdbcOperations()`
- `getNamedParameterJdbcOperations()`

are ALL present on which of the following classes?

Correct Answer	Your Selection	Answer	Explanation
✓	○	SimpleJdbcTemplate	CORRECT: This class provides all of these operations. Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/jdbc/core/simple/SimpleJdbcTemplate.html
	○	JdbcTemplate	INCORRECT: Although the methods queryForInt(args) and queryForList(args) are present on this class, getJdbcOperations() does not exist on this class which already implements JdbcOperations. This method exists on the wrapper classes SimpleJdbcTemplate and NamedParameterJdbcTemplate to get to the underlying JdbcTemplate. Similary getNamedParameterJdbcOperations() is not on this class. Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/jdbc/core/JdbcTemplate.html
	○	NamedParameterJdbcTemplate	INCORRECT: Although the methods getJdbcOperations(), queryForInt(args) and queryForList(args) are present on this class, getNamedParameterJdbcOperations() does not exist. This class implements NamedParameterJdbcOperations. This method exists on the class SimpleJdbcTemplate only. Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/jdbc/core/namedparam/NamedParameterJdbcTemplate.html
	○	All of the above	INCORRECT: Only SimpleJdbcTemplate has ALL of the methods.

Question :17 **Email**

Which of the following steps is INVALID regarding the integration of Spring's email functionality in to your application?

Correct Answer	Your Selection	Answer	Explanation
	○	Add mail.jar (JavaMail) and activation.jar (JAF) in to your classpath.	INCORRECT: The question required you to select an INVALID step. This step is valid. These jars are needed to use the Spring mail functionality. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/mail.html#mail-introduction
	○	Define a JavaMailSenderImpl bean that takes the SMTP settings (host, authentication etc.)	INCORRECT: The question required you to select an INVALID step. This step is valid. This implementation of the MailSender interface is the one to use when sending emails from Spring. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/mail.html#mail-usage-simple
✓	○	Define a MailTemplate bean which will be the API that simplifies usage to the underlying JavaMail API.	CORRECT: This step is invalid. There is no such class as MailTemplate. The MailSender implementation: JavaMailSenderImpl is essentially the API used to wrap calls to send messages out to the SMTP server.
	○	Use the SimpleMailMessage API to send a basic email.	INCORRECT: The question required you to select an INVALID step. This step is valid. SimpleMailMessage is an object representation of an email and allows you to set the to, cc, bcc, subject and text etc. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/mail.html#mail-usage-simple

Question :18 JMS

Which of the following statement is TRUE regarding the `JmsTemplate`?

Correct Answer	Your Selection	Answer	Explanation
	○	It should be configured using "prototype" scope to ensure one instance is created every time it is used to avoid thread issues.	INCORRECT: JmsTemplate is thread-safe and therefore a singleton instance is sufficient. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jms.html#jms-jmstemplate
✓	○	The JmsTemplate requires a reference to a ConnectionFactory.	CORRECT: The ConnectionFactory is the entry point for working with JMS. It is used to create connections with the JMS provider and encapsulates various configuration parameters, many of which are vendor specific such as SSL configuration options. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jms.html#jms-connections
	○	JmsTemplate supports the receiving of asynchronous message.	INCORRECT: The receive method is blocking. You should use MessageListeners for asynchronous message processing. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jms.html#jms-connections
	○	JmsTemplate fully supports conversion between Java objects and JMS messages.	INCORRECT: The MessageConverter API is provided for this purpose. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jms.html#jms-msg-conversion

Question :19 Database

To use an embedded HSQL database in Spring, that runs a script called `skillguru.sql`, which steps could you use from the below?

Correct Answer	Your Selection	Answer	Explanation
✓	☐	Use the "embedded-database" tag in the spring-jdbc namespace: <jdbc:embedded-database id="dataSource"> <jdbc:script location="classpath:skillguru.sql"/> </jdbc:embedded-database>	CORRECT: This is a valid approach using Spring XML to declare an embedded database Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jdbc.html#jdbc-embedded-database-xml
	☐	Use a programmatic approach as below, i.e. EmbeddedDatabaseBuilder builder = new EmbeddedDatabaseBuilder(); EmbeddedDatabase db = builder.setType(H2).addScript("skillguru.sql").build(); // do stuff with the db (EmbeddedDatabase extends javax.sql.DataSource) db.shutdown()	INCORRECT: The code is legal and would work - but it would use the H2 database, not the HSQL database.
	☐	Use the "embedded" tag in the spring-jdbc namespace: <jdbc:embedded id="dataSource"> <jdbc:script location="classpath:skillguru.sql"/> </jdbc:embedded>	INCORRECT: There is no "embedded" tag in the jdbc namespace. The correct tag is "embedded-database".
✓	☐	Use a programmatic approach as below, i.e. EmbeddedDatabaseBuilder builder = new EmbeddedDatabaseBuilder(); EmbeddedDatabase db = builder.addScript("skillguru.sql").build(); // do stuff with the db (EmbeddedDatabase extends javax.sql.DataSource) db.shutdown()	CORRECT: This is valid approach for setting up an embedded database programmatically. The database type is not specified which defaults to HSQL. Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/jdbc/datasource/embedded/EmbeddedDatabaseBuilder.html Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jdbc.html#jdbc-embedded-database-java

Question :20 Container & Test

If a class has the following methods...

```
@PreDestroy public shutDown() {

   System.out.println("shutDown ");

}

public closeMe() {

   System.out.println("closeMe");

}
```

...and the bean is configured with... `destroy-method="closeMe"`

Assuming the container is configured to process annotations correctly - what would be the expected order of the log output?

Correct Answer	Your Selection	Answer	Explanation
	○	None - a RuntimeException would be thrown as it is not legal to define more that one lifecycle destroy mechanism on a single bean	INCORRECT: You may define multiple lifecycle mechanisms - even all for shutdown, or initialization - on a single bean definition. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-factory-lifecycle-combined-effects
	○	"closeMe" only. This is because if you define a destroy-method this takes precedence over any other destroying lifecycle mechanism.	INCORRECT: Multiple lifecycle mechanisms are supported. However, if the destroy-method had referred to the same method that was annotated then this method would only execute once. In this example it configured a separate method so the statements will both execute. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-factory-lifecycle-combined-effects
✓	○	"shutDown" then "closeMe" are printed.	CORRECT: @PreDestroy takes precedence over the destroy-method, but both are executed. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-factory-lifecycle-combined-effects
	○	"closeMe" then "shutDown" are printed.	INCORRECT: See above for reason.

Question :21 Container & Test

In the following code excerpt...

```
// create and configure beans

ApplicationContext context = new ClassPathXmlApplicationContext(new String
[] {"services.xml", "daos.xml"});

// retrieve configured instance

CustomerService service = (CustomerService )context.getBean
("customerService");
```

How has this approach been improved in Spring 3.0?

Select 2 answers.

Correct Answer	Your Selection	Answer	Explanation
	☐	You no longer programmatically need to load an ApplicationContext - it is done purely through configuration.	INCORRECT: There are numerous reasons why you may need to instantiating your Spring IoC container, and this API is still valid. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-factory-instantiation
✓	☐	A new overloaded constructor on ClassPathXmlApplicationContext that uses varargs reduces code and enables you to construct an instance like so... ApplicationContext context = new ClassPathXmlApplicationContext ("services.xml", "daos.xml");	CORRECT: Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/context/support/ClassPathXmlApplicationContext.html#ClassPathXmlApplicationContext (java.lang.String...)
✓	☐	A new overloaded getBean() method that takes a type reduces code and enables you to retrieve your bean like so... // retrieve configured instance CustomerService service = context.getBean("customerService", CustomerService .class);	CORRECT: Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/beans/factory/BeanFactory.html#getBean (java.lang.Class)
	☐	ClassPathXmlApplicationContext and FileSystemXmlApplicationContext have both been deprecated and replaced with a single simplified class that can utilise both approaches to load a container.	INCORRECT: This is not true - these two classes remain as distinct and valid ways of loading up a container programmatically.

Question :22 Container & Test

Which of the following annotations is *NOT* component scanned by default?

Correct Answer	Your Selection	Answer	Explanation
	○	@Repository	INCORRECT: @Repository is a specialization of @Component for a more specific use case, for example, in the persistence layer. Because it extends @Component it is a valid candidate for component-scanning. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-stereotype-annotations
	○	@Service	INCORRECT: @Serviceis a specialization of @Component for a more specific use case, for example, in the service layer. Because it extends @Component it is a valid candidate for component-scanning. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-stereotype-annotations
✓	○	@Bean	CORRECT: This is not a @Component or subclass and therefore is not a candidate for component-scanning.
	○	Your custom annotation that is itself annotated with @Component	INCORRECT: See final answer for explanation.
	○	@Component	INCORRECT: This is the key annotation type that is may also be extended and will be found when classpath scanning. Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/stereotype/Component.html

Question :23 Web MVC

The `@RequestMapping` annotation may be used to map URIs to Java code in Spring MVC.

Which of the following 2 statements regarding this annotation are CORRECT?

Correct Answer	Your Selection	Answer	Explanation
	☐	It may be applied only to a method and not a class.	INCORRECT: See answer option below
✓	☐	It may be applied to a method and a class	CORRECT: If defined on a class then this forms the base URI of the request. Any annotated methods will map to URIs that are appended to the base URI.
	☐	You may define the http request method (GET or POST) as well as a cookie name to access this part of the http request.	INCORRECT: See answer option below.
✓	☐	You may define the http request method (GET or POST) but NOT any cookie details.	CORRECT: You may define the request method like so... @RequestMapping(method = RequestMethod.GET) The @CookieValue annotation is provided for the last feature, i.e. public void logSessionId(@CookieValue ("JSESSIONID") String cookie)

Question :24 AOP

With the following pointcut expression in mind...

```
execution(* set*(..))
```

Which of the following methods would be matched?

Correct Answer	Your Selection	Answer	Explanation
	○	private void setFoo(String bar)	INCORRECT: All are correct
	○	public void setBar()	INCORRECT: All are correct
	○	public String setFoo(int bar)	INCORRECT: All are correct
	○	public void set(String foo, int bar)	INCORRECT: All are correct
✓	○	All of the above	CORRECT: This pattern matches the execution of any method with a name beginning with "set" including "set" alone. Any number of arguments are allowed (0 to many). Any return type. Any access modifier. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/aop.html#aop-pointcuts-examples

Question :25 JMX

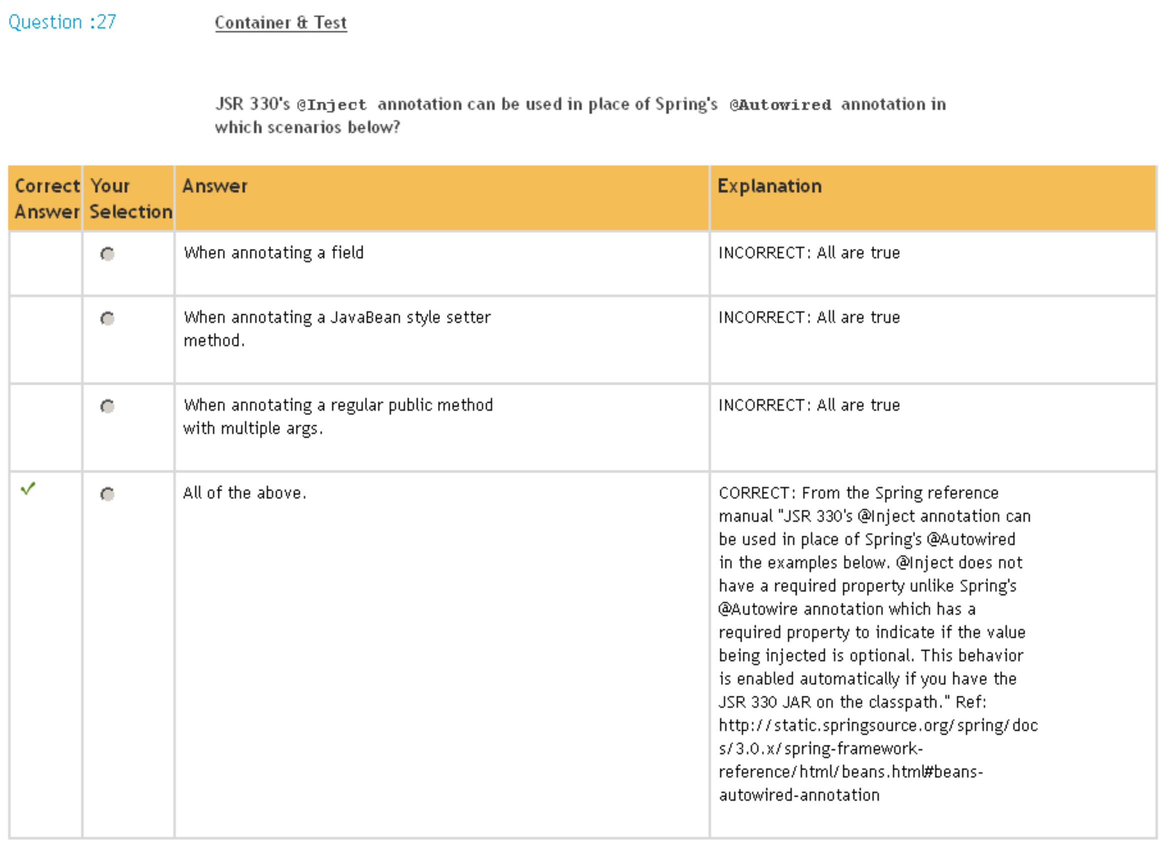
Assemblers configure your application on the rules needed to decide which components are exported as managed resources. Which of the following is NOT a valid assembler in Spring 3.0?

Correct Answer	Your Selection	Answer	Explanation
	○	MetadataMBeanInfoAssembler	INCORRECT: This assembler is used to export beans that are annotated. The question required you to select an assembler that is not valid. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jmx.html#jmx-interface-autodetect
	○	MethodNameBasedMBeanInfoAssembler	INCORRECT: This assembler is used to export beans based on a list of method names. The question required you to select an assembler that is not valid. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jmx.html#jmx-interface-methodnames
✓	○	AttributeBasedMBeanInfoAssembler	CORRECT: There is no such assembler in Spring.
	○	InterfaceBasedMBeanInfoAssembler	INCORRECT: This assembler is used to export beans based on a list of interfaces. The question required you to select an assembler that is not valid. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jmx.html#jmx-interface-java

Question :26 Container & Test

The default bean scope in Spring is 'singleton' ?

Correct Answer	Your Selection	Answer	Explanation
✓	○	TRUE	CORRECT: The following 2 definitions both resolve to the same result, i.e. a singleton instance of accountService inside the Spring container. <bean id="accountService" class="com.foo.DefaultAccountService"/> <bean id="accountService" class="com.foo.DefaultAccountService" scope="singleton"/>
	○	FALSE	INCORRECT: See answer above



Question :27 **Container & Test**

JSR 330's `@Inject` annotation can be used in place of Spring's `@Autowired` annotation in which scenarios below?

Correct Answer	Your Selection	Answer	Explanation
	○	When annotating a field	INCORRECT: All are true
	○	When annotating a JavaBean style setter method.	INCORRECT: All are true
	○	When annotating a regular public method with multiple args.	INCORRECT: All are true
✓	○	All of the above.	CORRECT: From the Spring reference manual "JSR 330's @Inject annotation can be used in place of Spring's @Autowired in the examples below. @Inject does not have a required property unlike Spring's @Autowire annotation which has a required property to indicate if the value being injected is optional. This behavior is enabled automatically if you have the JSR 330 JAR on the classpath." Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-autowired-annotation

Question :28 Web MVC

Consider the following web.xml fragment...

```
<servlet>

   <servlet-name>ikoko</servlet-name>

   <servlet-class> <THE CONTROLLER SERVLET> </servlet-class>

   <load-on-startup>1</load-on-startup>

</servlet>

<servlet-mapping>

   <servlet-name>ikoko</servlet-name>

    <url-pattern>/skillguru/*</url-pattern>

</servlet-mapping>
```

In order to complete configuration of this Spring MVC application - what should be used in place of <THE CONTROLLER SERVLET> ?

And what is the name of the Spring XML file that will, by default, be searched for?

Correct Answer	Your Selection	Answer	Explanation
	○	org.springframework.web.servlet.DispatcherServlet and applicationContext.xml	INCORRECT: See correct answer for explanation
✓	○	org.springframework.web.servlet.DispatcherServlet and ikoko-servlet.xml	CORRECT: From the reference manual... "Spring's web MVC framework is, like many other web MVC frameworks, request-driven, designed around a central servlet that dispatches requests to controllers and offers other functionality that facilitates the development of web applications. Spring's DispatcherServlet however, does more than just that. It is completely integrated with the Spring IoC container and as such allows you to use every other feature that Spring has." Custom servlets are not necessary as Spring provides the DispatcherServlet for this purpose. Your custom behaviour is placed in your Controller classes. From the reference manual... "Upon initialization of a DispatcherServlet, the framework looks for a file named [servlet-name]-servlet.xml in the WEB-INF directory of your web application and creates the beans defined there, overriding the definitions of any beans defined with the same name in the global scope." Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/mvc.html#mvc-servlet
	○	Your custom servlet and applicationContext.xml	INCORRECT: See correct answer for explanation
	○	Your custom servlet and ikoko-servlet.xml	INCORRECT: See correct answer for explanation

Question :29 Transactions

The `TransactionDefinition` interface specifies Isolation, Propagation, Timeout, and Read-only status?

Correct Answer	Your Selection	Answer	Explanation
✓	○	TRUE	CORRECT: These settings are defined on this interface and reflect standard transactional concepts. Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/transaction/TransactionDefinition.html Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/transaction.html#transaction-strategies
	○	FALSE	INCORRECT: See above answer

Question :30 SpEL

The expression language supports the Ant $ syntax for expressing values in your application context XML file/s or annotations?

Correct Answer	Your Selection	Answer	Explanation
	○	TRUE	INCORRECT: See answer below
✓	○	FALSE	CORRECT: SpEL uses the # character to denote SpEL expressions in the application context XML file/s or annotations. This is used when defining BeanDefinitions. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/expressions.html#expressions-beandef

Question :31 Container & Test

Which of the 2 following are suitable approaches for unit testing using Spring.

Correct Answer	Your Selection	Answer	Explanation
✓	☐	Use ModelAndViewAssert to unit test your Spring MVC controllers.	CORRECT: Together with other classes in the org.springframework.mock.web package, this class is a useful means of providing unit testing in to your project. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/testing.html#mock-objects
	☐	Use the @ContextConfiguration annotation to define your application context files.	INCORRECT: This is a valid annotation for doing this purpose - but you should not use the Spring container, and therefore your application context files, when unit testing. This approach is suitable for integration testing.
	☐	Make your test class extend AbstractDependencyInjectionSpringContextTests	INCORRECT: There are 2 reasons why this class is not suitable. 1) It has been deprecated in Spring 3.0 in favor of using the listener-based test context framework. 2) This class (like the annotation in the previous answer) is intended for integration-testing and not unit-testing. Unit-testing should not utilise the Spring container that this legacy class provides access to. Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/test/AbstractDependencyInjectionSpringContextTests.html Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/testing.html#unit-testing
✓	☐	You can use numerous classes from the org.springframework.mock.web package or even dynamic mock objects such as EasyMock or MockObjects, to provide mocked collaborators to the classes you wish to test.	CORRECT: Spring provides numerous classes to mock out the Servlet API, or you can even use external mocking APIs, although the documentation refers to these being less convenient. REF: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/testing.html#mock-objects-servlet

Question :32 EJB

Study the four statements below...

1. Spring provides support for implementing EJBs
2. Spring provides support for accessing EJBs
3. Spring provides support for EJB 2.x and EJB 3.x
4. Spring simplifies JNDI lookups.

Which of these are true ?

Correct Answer	Your Selection	Answer	Explanation
	○	1, 2 & 3	INCORRECT: Statement 4 is also true.
	○	1, 2 & 4	INCORRECT: Statement 3 is also true.
	○	2, 3 & 4	INCORRECT: Statement 1 is also true.
✓	○	1, 2, 3 & 4	CORRECT: All four statements are true. In this respect Spring provides comprehensive support for accessing, implementing and simplifying both EJB 2.x and EJB3.x Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/ejb.html

Question :33 JCA

A CCI, or Common Client Interface, is an interface that an application can use to interact with an EIS (Enterpise Information System).

Consider the following statements regarding Spring's support for JCA...

1. The base resource to use JCA CCI is the `ConnectionFactory` interface.
2. You may directly instantiate a `ConnectionFactory`.
3. The `CciTemplate` simplifies the use of CCI since it handles the creation and release of resources.
4. It is not possible to use global transactions when a connector is used in non-managed mode.

Which of these statements are true?

Correct Answer	Your Selection	Answer	Explanation
	○	1, 2, 3	INCORRECT: 2 is FALSE. 1,3,4 are TRUE.
	○	1, 2, 4	INCORRECT: 2 is FALSE. 1,3,4 are TRUE.
✓	○	1, 3, 4	CORRECT: 1. The connector you use must provide an implementation of the ConnectionFactory interface. You then inject it into your components. These components can either be coded against the plain CCI API or utilise Spring's support classes for CCI access (e.g. CciTemplate). 2. You cannot directly instantiate a specific ConnectionFactory. You need to go through the corresponding implementation of the ManagedConnectionFactory interface for your connector. This interface is part of the JCA SPI specification. 3. CciTemplate simplifies the use of CCI since it handles the creation and release of resources. It is analogous to JdbcTemplate in this respect. 4. When you use a connector in non-managed mode, you can't use global transactions because the resource is never enlisted / delisted in the current global transaction of the current thread. The resource is simply not aware of any global Java EE transactions that might be running. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/cci.html
	○	All four statements are true	INCORRECT: 2 is FALSE. 1,3,4 are TRUE.

Question :34 Transactions

If a method is given a propagation type of NOT_SUPPORTED. What will happen in the event that the method is run within a transaction from the calling method?

Correct Answer	Your Selection	Answer	Explanation
	○	It throws an exception.	INCORRECT: This describes the propagation attribute of NEVER. Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/transaction/annotation/Propagation.html
	○	It supports the current transaction.	INCORRECT: This describes the propagation attribute of REQUIRED. Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/transaction/annotation/Propagation.html
	○	It executes within a nested transaction	INCORRECT: This describes the propagation attribute of NESTED. Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/transaction/annotation/Propagation.html
✓	○	It suspend the current transaction.	CORRECT: This describes the propagation attribute of NOT_SUPPORTED. Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/transaction/annotation/Propagation.html

Question :35 Web MVC

Which of the following statements is FALSE regarding multipart handling / file uploading in forms in Spring?

Correct Answer	Your Selection	Answer	Explanation
✓	○	By default Spring does multi-part handling	CORRECT: You have to configure multipart handling in order for the DispatcherServlet to deal with these kinds of http requests. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/mvc.html#mvc-multipart
	○	You enable Spring multipart handling by adding a multipart resolver to the web application's context.	INCORRECT: This statement is true. The question required you to indentify the FALSE statement.
	○	If you configure CommonsMultipartResolver, you need to add commons-fileupload.jar to your classpath.	INCORRECT: This statement is true. The question required you to indentify the FALSE statement.
	○	When the Spring DispatcherServlet detects a multi-part request, it activates the resolver that has been declared in your context and hands over the request.	INCORRECT: This statement is true. The question required you to indentify the FALSE statement.

Question :36 AOP

The `ProxyFactoryBean` chooses to create one of either a JDK or CGLIB-based proxy for a particular target object (that is to be proxied).

Which of the following is NOT a consideration when choosing between them?

Correct Answer	Your Selection	Answer	Explanation
✓	○	Whether performance is an issue	CORRECT: Performance should NOT be considered. Although CGLIB proxying is considered to be marginally less performant than JDK (or interface) based proxying - the difference is marginal and could change in the future. The official documentation recommends that performance should NOT be a consideration when choosing between proxying approaches. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/aop-api.html#aop-pfb-proxy-types
	○	Whether the target classes, or their methods, are declared as final	INCORRECT: The question required you to select the factor that should NOT be considered. This is a genuine factor to consider as you cannot use CGLIB with final classes/methods as this approach relies on subclassing to proxy the target class.
	○	Whether the target classes implement interfaces	INCORRECT: The question required you to select the factor that should NOT be considered. This is a genuine factor to consider as you cannot use the JDK approach of proxying with classes that do not implement interfaces. No interfaces means JDK proxying isn't possible.
	○	Whether you can modify the source code of the target classes	INCORRECT: The question required you to select the factor that should NOT be considered. This is a genuine factor to consider as if you cannot modify the source code you will be restricted in which approach you can take. The above two answers explain more background to this.

Question :37 Container & Test

`MockTest` has an `initialize()` method, but no `init()` method.

`SpringMockTest` does *not* have an `init()` method.

...and they are configured like so...

```
<beans default-init-method="init">

  <bean id="mockTest" class="com.foo.MockTest" init-method="initialize" />

  <bean id="springMockTest" class="com.foo.SpringMockTest" />

</beans>
```

What would you expect to happen at runtime?

Correct Answer	Your Selection	Answer	Explanation
	○	An exception is thrown because the beans definition requires an init() method which is absent from both MockTest and SpringMockTest.	INCORRECT: When a default-init-method is defined it will be called on any beans that provide an implementation of the relevant method. Otherwise the bean is ignored. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-factory-lifecycle-default-init-destroy-methods
	○	An exception is thrown because the MockTest bean defines its own init-method which is not legal when also defining a top-level "default-init-method"	INCORRECT: You can combine lifecycle mechanisms in one bean - they do not cause an exception. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-factory-lifecycle-combined-effects
	○	An exception is thrown because the "springMockTest" does not have an init() method and this is required by the top-level "default-init-method" beans definition. "mockTest" is okay because it provides the container with an alternative lifecycle mechanism via its "init-method" attribute	INCORRECT: For both of the reasons above - this answer is wrong.
✓	○	None of the above.	CORRECT: None of the above are correct. Spring provides a flexible mechanism for multiple lifecycle approaches in the same container and even on the same bean. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-factory-lifecycle-default-init-destroy-methods

Question :38

AOP -

What is wrong with the following code excerpt that uses PCDs (pointcut definitions)?

```
@Pointcut("execution(public * *(..))")

private void anyPublicOperation() {}

@Pointcut("within(com.ikoko..*)")

private void inIkoko() {}

@Pointcut("anyPublicOperation() && inIkoko()")

private void examOperation() {}
```

Correct Answer	Your Selection	Answer	Explanation
	○	In the final pointcut t is not legal to use the && operator within an expression.	INCORRECT: A pointcut expression may utilise '&&', '||' and '!' to combine with another expression
	○	The 2nd pointcut has an erroneous two dots and should be a single dot	INCORRECT: 2 dots are legal - it signifies that any class within the com.ikoko package OR any sub-package. A single dot would match any class in the immediate com.ikoko package only.
	○	In the final pointcut the expression refers to other pointcuts by name. This is not legal.	INCORRECT: It is possible to refer to pointcut expressions by name.
✓	○	There is nothing wrong.	CORRECT: The code excerpt shows a completely legal use of pointcut definitions. The above explanations cover each one specifically. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/aop.html#aop-pointcuts-combining

Question :39 JMX

Which core class in Spring's JMX framework is responsible for taking Spring beans and registering them with a JMX `MBeanServer`?

Correct Answer	Your Selection	Answer	Explanation
	○	JMXExporter	INCORRECT: There is no such class
✓	○	MBeanExporter	CORRECT: This is the core class in Spring's JMX framework responsible for taking Spring beans and registering them with a JMX MBeanServer Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jmx.html#jmx-exporting
	○	ManagedResource	INCORRECT: This is a class level annotation indicating that instances of an annotated class are to be registered with a JMX server. Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/jmx/export/metadata/ManagedResource.html
	○	MBeanServerFactoryBean	INCORRECT: This class is responsible for creating an MBeanServer. Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/jmx/support/MBeanServerFactoryBean.html

Question :40 Container & Test

Spring 3.0 introduces more support for annotation based configuration. Consider the following configuration.

```
<bean id="springExamService"
class="com.skillguru.services.SpringExamServiceImpl"/>
```

What would be the equivalent annotated code?

Correct Answer	Your Selection	Answer	Explanation

	○	@BeanFactory
		public class AppConfig {
		  @Bean public SpringExamService springExamService() {
		    return new SpringExamServiceImpl();
		  }
		}
			INCORRECT

✓	○	@Configuration
		public class AppConfig {
		  @Bean public SpringExamService springExamService() {
		    return new SpringExamServiceImpl();
		  }
		}
			CORRECT: Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-java-basic-concepts

	○	@Configuration
		public class AppConfig {
		  @ConfigurationBean public SpringExamService springExamService() {
		    return new SpringExamServiceImpl();
		  }
		}
			INCORRECT

	○	There is no annotated equivalent for that configuration.	INCORRECT

Question :41

Container & Test

Consider the following two configuration fragments whereby the property 'answer' is of type `java.lang.String:`

```
<bean class="skillGuruBean1">

  <property name="answer" value=""/>

</bean>

<bean class="skillGuruBean2">

  <property name="answer"><null/></property>

</bean>
```

Consider the following statements regarding the possible values of 'answer' in each bean...

1. skillGuruBean1.answer is a blank String or ""
2. skillGuruBean1.answer is null
3. skillGuruBean2.answer is a blank String or ""
4. skillGuruBean2.answer is null
5. skillGuruBean2 throws a `RuntimeException` upon initialization as there is no such tag as `<null />`

Which of the above statements are CORRECT?

Correct Answer	Your Selection	Answer	Explanation
	○	1 & 3	INCORRECT
✓	○	1 & 4	CORRECT: The above configuration is equivalent to the following Java code: skillGuruBean1.setAnswer(""); skillGuruBean2.setAnswer(null); Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-null-element
	○	1 & 5	INCORRECT
	○	2 & 4	INCORRECT
	○	2 & 5	INCORRECT

Question :42 **Dynamic Language Support**

The dynamic languages currently supported are:

Correct Answer	Your Selection	Answer	Explanation
	○	Groovy	INCORRECT: All of the options are supported.
	○	JRuby	INCORRECT: All of the options are supported.
	○	BeanShell	INCORRECT: All of the options are supported.
✓	○	All of the above	CORRECT: The dynamic languages currently supported are: JRuby 0.9 / 1.0; Groovy 1.0 / 1.5; BeanShell 2.0. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/dynamic-language.html

Question :43 Remoting

You have been tasked with exposing your hibernate application as a remote service. You will also be creating the client. Just as when you thought your lucky streak might run out - you have been blessed with a co-operative networks administrator who is willing to open up any ports that you need.

From the following technologies below which would you consider as a viable option?

1. Spring's HTTP invoker
2. RMI
3. Hessian
4. Burlap

Correct Answer	Your Selection	Answer	Explanation
✓	○	1 & 2	CORRECT: From the reference below the Spring reference documentation notes that the Hessian and Burlap technologies have limitations especially around the usage of Hibernate and lazy-initialized collections. In these instances they recommend the RMI and Http Invoker approaches. Seeing as there are no network port issues, RMI is still an option. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/remoting.html#remoting-considerations
	○	3 & 4	
	○	All four are viable technologies to use	
	○	None of the above are suitable.	

Question :44 Database -

Spring's `LocalSessionFactoryBean` supports the use of both traditional Hibernate mapping (hbm.xml) files AND `javax.persistence` (i.e. `@Entity`) annotated POJOs?

Correct Answer	Your Selection	Answer	Explanation
	○	TRUE	INCORRECT
✓	○	FALSE	CORRECT: It is possible to initialize Hibernate using javax.persistence annotations. However, you need to declare an AnnotationSessionFactoryBean and define a list of annotated classes. This class is a subclass of LocalSessionFactoryBean . REF: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/orm/hibernate3/annotation/AnnotationSessionFactoryBean.html

Question :45 Type Conversion

Spring 3.0 provides the `PropertyEditor` for performing thread-safe type conversion. This allows any Type to be converted to any other Type.

Correct Answer	Your Selection	Answer	Explanation
	○	TRUE	INCORRECT:
✓	○	FALSE	CORRECT: org.springframework.core.convert.ConversionService is the new Spring 3.0 interface introduced for the purpose of converting any object to any other type. It defines a method: <T> T convert (Object source, Class<T> targetType) ...which performs a thread-safe type conversion. Within a Spring container, this system can be used as an alternative to PropertyEditors to convert externalized bean property value strings to required property types. The public API may also be used anywhere in your application where type conversion is needed. PropertyEditors are for converting a String to an object and vice versa. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/validation.html#core-convert

Question :46 Container & Test

`ReflectionTestUtils` is a useful Spring API for usage in unit and integration test methods.

What feature does this API provide?

Correct Answer	Your Selection	Answer	Explanation
✓	○	To be able to set a non-public field or invoke a non-public setter method when testing code.	CORRECT: From the below link... " ReflectionTestUtils is a collection of reflection-based utility methods for use in unit and integration testing scenarios. There are often situations in which it would be beneficial to be able to set a non-public field or invoke a non-public setter method when testing code involving, for example: * ORM frameworks such as JPA and Hibernate which condone the usage of private or protected field access as opposed to public setter methods for properties in a domain entity. * Spring's support for annotations such as @Autowired and @Resource which provides dependency injection for private or protected fields, setter methods, and configuration methods." Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/test/util/ReflectionTestUtils.html
	○	A collection of assertions intended to simplify testing scenarios dealing with Spring Web MVC ModelAndView objects.	INCORRECT: This describes ModelAndViewAssert.java Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/test/web/ModelAndViewAssert.html
	○	Provides support for dependency injection and initialization of test instances.	INCORRECT: This describes DependencyInjectionTestExecutionListener.java Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/test/context/support/DependencyInjectionTestExecutionListener.html
	○	Provides functionality of the Spring TestContext Framework to standard JUnit 4.5+ tests by means of the TestContextManager and associated support classes and annotations.	INCORRECT: This describes SpringJUnit4ClassRunner.java Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/test/context/junit4/SpringJUnit4ClassRunner.html

Question :47 AOP

Which of the following 2 statements are true regarding the decisions you might need to make regarding the use of aspect oriented programming?

Correct Answer	Your Selection	Answer	Explanation
✓	☐	Business logic that contains transactional code should be considered as a candidate for using aspects.	CORRECT: Transactional code is a classic example of what is known as a "cross-cutting concern" and is a suitable candidate for encapsulating in to an advice. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/aop.html#aop-introduction
	☐	AspectJ is simpler than using Spring AOP.	INORRECT: Spring AOP is simpler as there is no need to use the AspectJ compiler / weaver into your development. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/aop.html#aop-spring-or-aspectj
✓	☐	If you have chosen to use Spring AOP, then you have a choice of @AspectJ or XML style.	CORRECT: Yes, these are the two styles you have to choose from if you have decided to use Spring AOP. It is confusing because you have chosen Spring AOP, as opposed to full AspectJ, yet within Spring AOP you still chose the @AspectJ or XML style. Notice that the style is an annotation style. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/aop.html#aop-ataspectj-or-xml
	☐	@AspectJ has two disadvantages: it does not encapsulate the implementation of the requirement in a single place. Secondly, @AspectJ is slightly more limited in what it can express than the XML style.	INCORRECT: Both these statements are the wrong way around. The XML style does not encapsulate the aspect information in the one place - it is in the XML and the Java code. The @AspectJ style offers more features than the more restrictive XML. For example, you may combine pointcuts when using @AspectJ. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/aop.html#aop-ataspectj-or-xml

Question :48 Remoting

Spring 3.0 provides support for exposing a bean as a JAX-WS service, via exporting it using the `SimpleJaxWsServiceExporter`.

This is a similar concept to using an `RmiServiceExporter` or a `HessianServiceExporter`.

Correct Answer	Your Selection	Answer	Explanation
✓	○	TRUE	CORRECT: One approach is to annotate a class and its methods with @WebService and @WebMethod, then register it with the JAX-WS engine using this exporter. Please note that this approach is limited to working in a standalone environment. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/remoting.html#remoting-web-services-jaxws-export-standalone
	○	FALSE	INCORRECT

Question :49 JMX

What is wrong with the following annotated code?

```
public class Customer {

  @ManagedResource

  private String name;

  @ManagedAttribute

  public void setName(String name) {

    this.name = name;

  }

  public String getName() {

    return name;

  }

  @ManagedOperation

  public String getNameFormatted() {

    return name.toUpperCase();

  }

}
```

Correct Answer	Your Selection	Answer	Explanation
✓	○	@ManagedResource should be applied to a class only	CORRECT: This denotes a class as being exportable as a managed bean. It should not be used anywhere else. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jmx.html#jmx-interface-metadata-types
	○	@ManagedAttribute should be on a property / field only not on a setter.	INCORRECT:It is okay to use this annotation on the getters and setters of a class.
	○	@ManagedOperation should not be applied to this method as it is not a JavaBean style getter or setter.	INCORRECT: It is okay to use @ManagedOperation on a method that is not a getter or setter.
	○	None of these - the annotated bean is correct	INCORRECT: See the first answer for the problem with this code.

Question :50 General

Which of the following is NOT a new feature introduced in Spring 3.0?

Correct Answer	Your Selection	Answer	Explanation
	○	Spring Expression Language	INCORRECT: Spring Expression Language was introduced in Spring 3.0. The question requires you to select a feature that was NOT introduced. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/expressions.html
✓	○	SimpleJdbcTemplate updated to support Java 5.0 features	CORRECT: This class was already present in Spring 2.x and already supported Java 5.0 features such as generics, varargs and auto-boxing. In Spring 3.0, the original JdbcTemplate also now supports Java 5-enhanced syntax with generics and varargs. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jdbc.html#jdbc-SimpleJdbcTemplate
	○	Early support for Java EE6	INCORRECT: This was introduced in Spring 3.0. Support for asynchronous method invocations via the new @Async annotation (or EJB 3.1's @Asynchronous annotation). The question requires you to select a feature that was NOT introduced. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/new-in-3.html#new-in-3-features-overview Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/scheduling.html#scheduling-annotation-support-async
	○	REST supported enhanced	INCORRECT: This was introduced in Spring 3.0. The question requires you to select a feature that was NOT introduced. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/remoting.html#rest-client-access
	○	Embedded database support	INCORRECT: This was introduced in Spring 3.0. The org.springframework.jdbc.datasource.embedded package provides support for embedded Java database engines such as HSQL, H2, and Derby. The question requires you to select a feature that was NOT introduced. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jdbc.html#jdbc-embedded-database-support