


<http://www.skill-guru.com/test/81/core-spring-3.0-certification-mock>

Core Spring 3.0 Certification Mock

SpringSource have recently updated Core Spring Certification to version 3.0 of the framework. This exam is now updated accordingly. This mock test aims to be the closest to the real exam on the web. First 10 questions free! Bargain at \$1.99

Category	Certification / Mock test
Keywords	Spring, SpringSource, Mock, Examination, Core, VMWare, Exam, Test, 3.0, 2.5.6, 2.5, Certification
Tags	spring java certification 3.0 core springsource
Questions	10- Free questions 40 - Paid questions
Test takers	1944
Average score	53.66
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

Question :1

Spring 3.0 - Web & MVC - The following static methods exist on which Spring class?

Object `getOrCreateSessionAttribute(HttpSession session, String name, Class clazz)`
 String `extractFilenameFromUriPath(String uriPath)`
 Cookie `getCookie(HttpServletRequest request, String name)`

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	WebApplicationContext	INCORRECT: WebApplicationContext is a read-only interface to provide configuration for a web application. Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/web/context/WebApplicationContext.html
	<input type="radio"/>	WebApplicationContextUtils	INCORRECT: This class provides convenience methods for retrieving the root WebApplicationContext for a given ServletContext. This is useful for accessing a Spring context from within custom web views or Struts actions. Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/web/context/support/WebApplicationContextUtils.html
✓	<input type="radio"/>	WebUtils	CORRECT: WebUtils provides miscellaneous utilities for web applications. These are used by various framework classes. Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/web/util/WebUtils.html
	<input type="radio"/>	AbstractController	INCORRECT: These methods do not exist on this class. This is a convenient superclass for controller implementations, using the Template Method design pattern. It provides instance methods, not static utility methods. Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/web/servlet/mvc/AbstractController.html

Question :2

Spring 3.0 - Web & MVC - Spring 3.0 does NOT offer support for multi-part forms?

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	TRUE	
✓	<input checked="" type="radio"/>	FALSE	Spring's multipart support handles uploads of files in web apps. Multipart support is enabled with MultipartResolver objects. Spring gives you a MultipartResolver to use with Commons FileUpload. http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/mvc.html#mvc-multipart

Question :3

Spring 3.0 - Web & MVC - Which of the following statements regarding the DispatcherServlet is FALSE?

Correct Answer	Your Selection	Answer	Explanation
✓	<input checked="" type="radio"/>	The DispatcherServlet will default to looking up a configuration file called: applicationContext.xml unless explicitly over-riden in the configuration.	CORRECT: This statement about DispatcherServlet is FALSE. From the Spring reference documentation: "The framework will, on initialization of a DispatcherServlet, look for a file named [servlet-name]-servlet.xml in the WEB-INF directory of your web application and create the beans defined there (overriding the definitions of any beans defined with the same name in the global scope)." The other three statements regarding the DispatcherServlet are true. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/mvc.html#mvc-servlet
	<input type="radio"/>	The DispatcherServlet is really just a Servlet (subclass of HttpServlet), and as such is declared in the web.xml of a web application.	INCORRECT: This statement is TRUE. The answer requires you to pick the FALSE one.
	<input type="radio"/>	The DispatcherServlet is analogous to the "Front Controller" in the design pattern of the same name.	INCORRECT: This statement is TRUE. The answer requires you to pick the FALSE one.
	<input type="radio"/>	The DispatcherServlet will default to using the XmlWebApplicationContext to instantiate the context unless explicitly over-riden in the configuration.	INCORRECT: This statement is TRUE. The answer requires you to pick the FALSE one.

Question :4

Spring 3.0 - AOP - An "after returning advice" is a class that is either annotated with @AfterReturning (AspectJ) or implements the AfterReturningAdvice interface (Spring). This kind of advice will only execute if the method returns normally, i.e. with no exceptions. True/False?

Correct Answer	Your Selection	Answer	Explanation
✓	<input checked="" type="radio"/>	True	CORRECT: This statement is true. If an exception is thrown from the target bean then an 'after returning' advice will not get executed. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/aop.html#aop-advice-after-returning The developer should consider alternative advices if they wish to execute code after an exception is thrown from the target bean. If the requirement is to execute after the target bean executes in all circumstances then an after (finally) advice should be considered. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/aop.html#aop-advice-after-finally If the requirement is to only execute in the event of an exception then an 'after throwing' advice should be used. The latter will not execute if the target bean executes without exception. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/aop.html#aop-advice-after-throwing
	<input type="radio"/>	False	INCORRECT: See full explanation against the correct answer above.

Question :5

Spring 3.0 - Transactions - Is the following statement true or false?

Spring's default behavior for declarative roll back is only for unchecked exceptions.

Correct Answer	Your Selection	Answer	Explanation
✓	<input checked="" type="radio"/>	True	CORRECT: By default, checked exceptions that are thrown from a transactional method will not result in the transaction being rolled back. It is possible to customize this behavior though. For example if it is required to rollback in the event that the checked exception MyCheckedException is thrown, then add an attribute "rollback-for" in the "tx:method" declaration, i.e. <code><tx:method name="get" read-only="true" rollback-for="MyCheckedException"/></code> Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/transaction.html#transaction-declarative-rolling-back
	<input type="radio"/>	False	INCORRECT: See description of TRUE answer.

Question :6

Spring 3.0 - Container & Test - Given the following code and configuration what would be the expected log/console output?

Code...

```
public class Foo {
    @PostConstruct
    public void init() {
        System.out.print("Annotation-and-init-method ");
    }
}
```

Configuration...

```
<bean class="com.foo.Foo" init-method="init" />
```

Assume that the BeanPostProcessor for the annotation is correctly defined.

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	"Annotation-and-init-method" is written twice	INCORRECT: See correct answer for explanation.
✓	<input checked="" type="radio"/>	"Annotation-and-init-method" is written once	CORRECT: As of Spring 2.5, there are three options for controlling bean lifecycle behavior. It is legal to combine the three approaches at once. They are executed in the following order: 1. The @PostConstruct annotations 2. The InitializingBean callback interface with its method afterPropertiesSet() 3. The init-method attribute of the bean which allows for a custom init() method However, if two approaches are used for the same method then that method will only be executed once. In the above example the @PostConstruct annotation is used on the same method that is in the bean definition as being the init-method, i.e. init(). Same rules apply to the @PostDestroy, DisposableBean and destroy-method. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-factory-lifecycle-combined-effects
	<input type="radio"/>	Nothing is written in to the log. The initialization silently fails as it is not allowed to combine multiple lifecycle mechanisms for a bean on to a single method.	INCORRECT: See correct answer for explanation.
	<input type="radio"/>	A RuntimeException is thrown as it is not allowed to combine lifecycle mechanisms for a bean on to a single method.	INCORRECT: See correct answer for explanation.

Question :7

Spring 3.0 - Transactions - How do you get a UserTransaction?

Correct Answer	Your Selection	Answer	Explanation
✓	<input checked="" type="radio"/>	JNDI	CORRECT: A JTA UserTransaction normally needs to be sourced from JNDI. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/transaction.html#doe19051
	<input type="radio"/>	ApplicationContext	INCORRECT: See first answer and reference.
	<input type="radio"/>	Construct one directly	INCORRECT: See first answer and reference.
	<input type="radio"/>	By calling TransactionTemplate.getUserTransaction() ()	INCORRECT: There is no such method. See first answer and reference.

Question :8

Spring 3.0 - Container & Test - Given the following code and configuration what would be the expected log/console output?

```
Code
public class Foo implements InitializingBean {
    @Override
    public void afterPropertiesSet() {
        System.out.print("InitializingBean ");
    }
    @PostConstruct
    public void doStuff() {
        System.out.print("Annotation ");
    }
    public void init() {
        System.out.print("init-method ");
    }
}
```

Configuration...

```
<bean class="com.foo.Foo" init-method="init" />
```

?assume that the BeanPostProcessor for the annotation is correctly defined.

Correct Answer	Your Selection	Answer	Explanation
✓	<input checked="" type="radio"/>	"Annotation", followed by "InitializingBean" and finally "init-method"	CORRECT: As of Spring 2.5, there have been three options for controlling bean lifecycle behavior. It is legal to combine the three approaches at once. They are executed in the following order: 1. The @PostConstruct annotations 2. The InitializingBean callback interface with its method afterPropertiesSet() 3. The init-method attribute of the bean which allows for a custom init() method. Same order is applied to the @PostDestroy, DisposableBean and destroy-method . Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-factory-lifecycle-combined-effects
	<input type="radio"/>	"InitializingBean" followed by "Annotation" and finally "init-method"	INCORRECT: See the correct order defined in the correct answer description.
	<input type="radio"/>	Nothing is written out to the log as it silently fails because of clashing lifecycle implementations	INCORRECT: It is legal to use several lifecycle approaches to initialization.
	<input type="radio"/>	An exception is thrown because of clashing lifecycle implementations	INCORRECT: It is legal to use several lifecycle approaches to initialization.

Question :9

Spring 3.0 - Remoting - Which of the following statements is FALSE regarding Spring and EJB?

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	Typically code using EJBs depends on Service Locator or Business Delegate singletons, making it hard to test.	
✓	<input type="radio"/>	Spring prevents you from using EJBs as the two technologies are in conflict with each other.	This is the correct answer as it is FALSE. It is a myth that Spring and EJB are diametrically opposed to one another. While this has some origins in Springs simpler configuration and less verbose code, Spring can be used to compliment EJB to provide a rich and powerful server-side solution. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/ejb.html#ejb-introduction
	<input type="radio"/>	Spring makes it much easier to access EJBs and implement EJBs.	
	<input type="radio"/>	Spring supports the older 2.x spec of EJB as well as the newer more POJO orientated EJB 3.x spec.	

Question :10

Spring 3.0 - Container & Test - Which of the following statements regarding the @Autowired annotation is correct?

Correct Answer	Your Selection	Answer	Explanation
✓	<input type="radio"/>	The @Autowired annotation may be applied to methods with arbitrary names (not just setters) and/or multiple arguments, for example: @Autowired prepareData(Foo foo, Bar bar)	CORRECT: The @Autowired annotation may be applied to methods with arbitrary names and/or multiple arguments as well as the classic setter methods. Remember that @Autowired may be applied to constructors, methods or fields. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-autowired-annotation
	<input type="radio"/>	The @Autowired annotation can only be applied to classic JavaBean style setter methods with single arguments, for example: @Autowired setFoo(Foo foo)	

Question :11

Spring 3.0 - Container & Test - Which of the following is true regarding the usage of the @Autowired annotation?

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	Applying @Autowired to a field is not legal and will result in a RuntimeException.	INCORRECT: @Autowired may be applied to a field.
✓	<input checked="" type="radio"/>	@Autowired may be applied to either a field, or the setter of a field.	@Autowired may be applied to: 1. setter methods, 2. methods with arbitrary names and/or multiple arguments, 3. constructors 4. fields, 5. arrays (whereby all beans of a particular type from the ApplicationContext are injected), 6. typed collections and 7. typed maps (as long as the key is a java.lang.String) Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-autowired-annotation
	<input type="radio"/>	Applying @Autowired to a field is not legal and will result in a checked exception.	INCORRECT: @Autowired may be applied to a field.
	<input type="radio"/>	@Autowired may be applied to setter methods only.	INCORRECT: There are various options regarding the application of @Autowired and not just fields and simple field setters.

Question :12

Spring 3.0 - Container & Test - If there are multiple instances of Foo defined in the applicationContext.xml and the following method is defined it will result in a RuntimeException .

@Autowired setFoo(List<Foo> foos)

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	True	
✓	<input checked="" type="radio"/>	False	CORRECT: The answer is false. It is possible to inject both arrays and collections (Set, List) from multiple definitions of a single type using this annotation. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-autowired-annotation

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	JdbcTemplate is not thread safe and therefore a separate instance should be created for each SQL statement that is executed.	INCORRECT: This statement is FALSE. The question requires you to select the TRUE statement. JdbcTemplate is thread safe and therefore shareable across SQL statements. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jdbc.html#jdbc-JdbcTemplate-idioms
✓	<input type="radio"/>	Multiple instances of JdbcTemplate are recommended if an application connects to multiple datasources.	CORRECT: This statement is TRUE. A JdbcTemplate is associated with a single datasource. Therefore it follows that if your application has more than one (SQL) datasource it would need additional JdbcTemplates configured for each one. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jdbc.html#jdbc-JdbcTemplate
	<input type="radio"/>	Your DAO implementation must extend Spring's JdbcDaoSupport if your class needs to use the JdbcTemplate.	INCORRECT: This statement is FALSE. The question requires you to select the TRUE statement. A DAO implementation does not necessarily have to extend JdbcDaoSupport in order to access a JdbcTemplate. One can be injected directly, or even a datasource can be injected allowing the DAO to create the instance itself. This approach is, therefore, just one approach for the developer to consider, but not mandatory. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jdbc.html#jdbc-JdbcTemplate-idioms
	<input type="radio"/>	JdbcTemplate is preferred to SimpleJdbcTemplate in a Java 5.0 environment or later.	INCORRECT: This statement is FALSE. The question requires you to select the TRUE statement. SimpleJdbcTemplate is the Java 5.0 version of JdbcTemplate that supports generics, varargs and auto-boxing. Not the other way around. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jdbc.html#jdbc-SimpleJdbcTemplate

Question :14

Spring 3.0 - JMX - Which of the following 2 statements regarding Spring JMX are TRUE ?

Note, that the terms "container" and "non-container" in the given options mean the following:

A "container" is a server such as Tomcat or Websphere.

A "non-container" environment is considered to be a standalone Java application.

Correct Answer	Your Selection	Answer	Explanation
✓	<input type="checkbox"/>	MBeans, or Spring JMX support may work in a non-container environment by configuring an MBeanServerFactoryBean.	CORRECT: This statement is TRUE. It is possible to get MBeans working in any environment. In non-container environments simply configure the MBeanServerFactoryBean and reference this in the MBeanExporter. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jmx.html#jmx-exporting-mbeanserver
	<input type="checkbox"/>	MBeans, or Spring JMX support do not work in a non-container environment. This is because these APIs required an MBeanServer only provided by container environments.	INCORRECT: This statement is FALSE. Please see the above answer and reference.
✓	<input type="checkbox"/>	Spring's JMX support allows for three different registration behaviours when the registration process finds that an MBean has already been registered under the same ObjectName.	CORRECT: This statement is TRUE. The default behaviour is to fail and throw an InstanceAlreadyExistsException when a duplicate objectName is registered. However, there are three behaviours (including the above) that can be configured in total: REGISTRATION_FAIL_ON_EXISTING, REGISTRATION_IGNORE_EXISTING, REGISTRATION_REPLACE_EXISTING Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jmx.html#jmx-exporting-registration-behavior
	<input type="checkbox"/>	Spring's JMX support allows for four different registration behaviours when the registration process finds that an MBean has already been registered under the same ObjectName.	INCORRECT: This statement is FALSE. Please see the above answer and reference.

Question :15

Spring 3.0 - Transactions - Which best describes the propagation attribute REQUIRED?

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	Create a new transaction, suspend the current transaction if one exists.	INCORRECT: This option describes the REQUIRES_NEW propagation attribute.
	<input type="radio"/>	Execute non-transactionally, suspend the current transaction if one exists.	INCORRECT: This option describes the NOT_SUPPORTED propagation attribute.
	<input type="radio"/>	Support a current transaction, throw an exception if none exists.	INCORRECT: This option describes the MANDATORY propagation attribute.
✓	<input type="radio"/>	Support a current transaction, create a new one if none exists.	CORRECT: This option describes the REQUIRED propagation attribute. Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/transaction/annotation/Propagation.html

Question :16

Spring 3.0 - Remoting - You have been tasked for creating a remote service using Java and Spring that a 3rd party .NET client will need to access across the internet. What would be the BEST technology choice from the list below?

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	RMI	INCORRECT: RMI is a Java-to-Java technology that would also not work across http.
	<input type="radio"/>	HttpInvoker	INCORRECT: HttpInvoker is a Java-to-Java technology.
✓	<input checked="" type="radio"/>	Web service	CORRECT: Web Services are the best solution because they work over http and completely decouple the server technology from the client technology. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/remoting.html#remoting-considerations
	<input type="radio"/>	Hessian or Burlap	INCORRECT: Hessian/Burlap have limited support for non-Java technologies.

Question :17

Spring 3.0 - AOP - Which of the following TWO statements are TRUE about the Advisor and Pointcut classes in the org.springframework.aop package?

Correct Answer	Your Selection	Answer	Explanation
	<input type="checkbox"/>	Advisor is composed of a ClassFilter and a MethodMatcher	INCORRECT - this is a description of a Pointcut , not an Advisor
	<input type="checkbox"/>	The Pointcut class holds an AOP advice (action to take at a joinpoint) and a filter determining the applicability of the advice.	INCORRECT - this is a description of an Advisor, not a Pointcut
✓	<input type="checkbox"/>	Pointcut is composed of a ClassFilter and a MethodMatcher	CORRECT: Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/aop-api.html#aop-api-pointcuts
✓	<input type="checkbox"/>	The Advisor class holds an AOP advice (action to take at a joinpoint) and a filter determining the applicability of the advice.	CORRECT: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/aop/Advisor.html

Question :18

Spring 3.0 - AOP - Which of the following statements are FALSE about an advice?

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	An advice is a Spring bean	INCORRECT: This statement is TRUE. The question requires you to select a FALSE statement. An Advice is declared as a normal Spring bean.
	<input type="radio"/>	An advice instance can be shared across all advised objects, or unique to each advised object. This corresponds to per-class or per-instance advice.	INCORRECT: This statement is TRUE. The question requires you to select a FALSE statement.
✓	<input type="radio"/>	Advices can be associated to both methods and fields	CORRECT: This statement is FALSE. "Spring AOP currently supports only method execution join points (advising the execution of methods on Spring beans). Field interception is not implemented, although support for field interception could be added without breaking the core Spring AOP APIs. If you need to advise field access and update join points, consider a language such as AspectJ." Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/aop.html#aop-introduction-spring-defn
	<input type="radio"/>	Spring provides several advice types out of the box, and is extensible to support arbitrary advice types.	INCORRECT: This statement is TRUE. The question requires you to select a FALSE statement.

Question :19

Spring 3.0 - Container & Test - What output would you expect in the console, AFTER start-up has completed, when running the client code below for the following beans?

Bean Configuration...

```
<bean id="newAccount" class="com.foo.NewAccount" scope="prototype" />
<bean id="accountService" class="com.foo.AccountService" scope="singleton"
autowire="byName"/>
```

Bean Code...

```
public class NewAccount {
    public NewAccount() {
        System.out.println("New account created ");
    }
}

public class AccountService {
    private NewAccount newAccount;
    public void setNewAccount(NewAccount newAccount) {
        this.newAccount = newAccount;
    }
}
```

Client code (assume valid applicationContext instance)

```
AccountService service1 = (AccountService)applicationContext.getBean("accountService");
AccountService service2 = (AccountService)applicationContext.getBean("accountService");
```

Correct Answer	Your Selection	Answer	Explanation
✓	<input checked="" type="radio"/>	"New account created" is NOT printed out after Spring container start-up.	CORRECT: During startup "New account created" will be printed out once because Spring by default will initialise all singleton beans (unless explicitly marked as lazy). However, after startup has completed nothing will be printed out. Note, even though NewAccount is declared as a prototype scope, it is owned by a singleton AccountService. Therefore only one instance will ever be created for the single owning bean. After startup any subsequent retrieval of the (already created) AccountService bean will return the same ("singleton") instance which has already been instantiated with the same NewAccount instance. This highlights the hazards of mixing scope types and the reference documentation gives an example of how to overcome this issue if you genuinely intend a singleton bean to return a brand new instance of a dependent (prototype) bean. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-factory-scopes-sing-prot-interaction Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-factory-method-injection
	<input type="radio"/>	"New account created" printed out once after Spring container start-up.	INCORRECT: The container has already created an instance of the prototype bean and injected it in to the AccountService during startup. Subsequent retrievals of the singleton AccountService accesss the same instance that was created before.
	<input type="radio"/>	"New account created" printed out twice after Spring container start-up.	INCORRECT: Same reason as above
	<input type="radio"/>	A RuntimeException is thrown when the container starts up because the newAccount property is not set.	INCORRECT: The autowire="byName" ensures that the "newAccount" bean is injected in to the "AccountService" bean.

Question :20

Spring 3.0 - JMS - Which of the following is a method on the class
org.springframework.jms.core.JmsTemplate?

Correct Answer	Your Selection	Answer	Explanation
✓	<input checked="" type="radio"/>	void convertAndSend(Destination destination, Object message)	CORRECT: This is a method on JmsTemplate. Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org.springframework.jms.core.JmsTemplate.html
	<input type="radio"/>	String buildExceptionMessage(JMSException ex)	INCORRECT: This is a method on JmsUtils Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org.springframework.jms.support.JmsUtils.html
	<input type="radio"/>	void commitIfNecessary(Session session)	INCORRECT: This is a method on JmsUtils Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org.springframework.jms.support.JmsUtils.html
	<input type="radio"/>	All of the above	INCORRECT: Only the first option is a method on JmsTemplate.

Question :21

Spring 3.0 - JMX - Which of the following is NOT an approach for defining a JMX assembler in Spring?

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	Method names	INCORRECT: This statement is a valid approach for defining an assembler via <code>MethodNameBasedMBeanInfoAssembler</code> . The question required to select an answer that was NOT suitable. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jmx.html#jmx-interface-methodnames
✓	<input checked="" type="radio"/>	Private field names	CORRECT: There is no explicit support for field names, although implicitly you could use getters and setters. Although public field names (and methods) are registered with <code>SimpleReflectiveMBeanInfoAssembler</code> . Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jmx.html#jmx-interface-assembler The main valid assemblers are: <code>MethodNameBasedMBeanInfoAssembler</code> , <code>InterfaceBasedMBeanInfoAssembler</code> and <code>MetadataMBeanInfoAssembler</code> .
	<input type="radio"/>	Interface names	INCORRECT: This statement is a valid approach for defining an assembler via <code>InterfaceBasedMBeanInfoAssembler</code> . Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jmx.html#jmx-interface-java The question required to select an answer that was NOT suitable.
	<input type="radio"/>	Metadata (or annotations)	INCORRECT: This statement is a valid approach for defining an assembler via <code>MetadataMBeanInfoAssembler</code> . The question required to select an answer that was NOT suitable. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jmx.html#jmx-interface-metadata

Question :22

Spring 3.0 - JDBC - Which of the following statements is FALSE regarding SimpleJdbcTemplate?

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	The SimpleJdbcTemplate class is a wrapper around the classic JdbcTemplate that takes advantage of Java 5 language features such as varargs and autoboxing.	INCORRECT: This statement is TRUE. The question requires you to select the FALSE statement. Please note that In Spring 3.0, the original JdbcTemplate now also supports Java 5-enhanced syntax with generics and varargs. However, the SimpleJdbcTemplate provides a simpler API.
✓	<input type="radio"/>	The SimpleJdbcTemplate adds support for programming JDBC statements using named parameters (as opposed to programming JDBC statements using only classic placeholder (?) arguments.	CORRECT: This statement is FALSE as this describes the NamedParameterJdbcTemplate. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jdbc.html#jdbc-SimpleJdbcTemplate
	<input type="radio"/>	SimpleJdbcTemplate class wraps a classic JdbcTemplate template and allows access to the underlying class via getJdbcOperations() method.	INCORRECT: This statement is TRUE. The question requires you to select the FALSE statement.
	<input type="radio"/>	SimpleJdbcDaoSupport is a convenience API that you may extend in order to gain access to the SimpleJdbcTemplate.	INCORRECT: This statement is TRUE. The question requires you to select the FALSE statement.

Question :23

Spring 3.0 - AOP - @AspectJ support is enabled in a Spring application by undertaking which of the following steps?

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	Declare <aspect-autoproxy/> in your application context configuration. There is no need to declare the aop namespace as this tag is part of the default Spring namespace in Spring 3.x.	INCORRECT: This tag is part of the aop namespace and must be declared.
✓	<input type="radio"/>	Declare <aop:aspectj-autoproxy/> in your application context configuration, and ensure the aop namespace is declared in the beans tag.	CORRECT: These are the two steps needed to configure your container for @AspectJ support. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/aop.html#aop-aj-configure
	<input type="radio"/>	AspectJ is an alternative to Spring AOP and therefore cannot be used inside a Spring container.	INCORRECT: AspectJ has origins separate from Spring but is well supported in the container although Spring does not use the AspectJ runtime.
	<input type="radio"/>	You do not need to declare anything as AspectJ is fully integrated, and by default supported, in Spring 3.x.	INCORRECT: Although AspectJ is supported, it is critical to first declare the <aop:aspect-autoproxy/>.

Question :24

Spring 3.0 - Container & Test - Which of the following TWO steps are required to implement a custom property-editor?

Correct Answer	Your Selection	Answer	Explanation
✓	<input type="checkbox"/>	Ensure your custom class extends PropertyEditorSupport and overrides void setAsText(String text) which will convert a String in to an instance of your target class.	CORRECT: This API is the one to extend when coding custom property editors. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/validation.html#beans-beans-conversion-customeditor-registration
	<input type="checkbox"/>	Ensure your custom class extends CustomEditorSupport and overrides void setAsText(String text) which will convert a String in to an instance of your target class.	INCORRECT: There is no class called CustomEditorSupport. Instead your class should extend PropertyEditorSupport.
	<input type="checkbox"/>	Use PropertyEditorConfigurer to register the customer property-editor with the Spring container.	INCORRECT: There is no class called PropertyEditorConfigurer. Instead use CustomEditorConfigurer.
✓	<input type="checkbox"/>	Use CustomEditorConfigurer to register the customer property-editor with the Spring container.	CORRECT: Use CustomEditorConfigurer to register the new PropertyEditor with the ApplicationContext. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/validation.html#beans-beans-conversion-customeditor-registration

Question :25

Spring 3.0 - Transactions - Which of the following statements about Spring's declarative transaction management is FALSE?

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	The Spring Framework enables declarative transaction management to be applied to any class, not merely special classes such as EJBs.	INCORRECT: This statement is TRUE. The question requires you to select a FALSE statement.
	<input type="radio"/>	Unlike EJB CMT and JTA approaches, the Spring Framework's declarative transaction management works with JDBC and certain ORM technologies.	INCORRECT: This statement is TRUE. The question requires you to select a FALSE statement.
✓	<input type="radio"/>	The Spring Framework supports propagation of transaction contexts across remote calls, unlike certain commercial application servers.	CORRECT: This statement is FALSE. Spring does not support propagation of transaction contexts across remote calls, unlike most commercial application servers. It is recommended by Spring to use EJB with the caveat that it is not normal to make transactions to span remote calls. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/transaction.html#transaction-declarative
	<input type="radio"/>	The Spring Framework offers declarative rollback rules; which does not occur in EJB.	INCORRECT: This statement is TRUE. The question requires you to select a FALSE statement. Both programmatic and declarative support for rollback rules is provided in Spring.

Question :26

Spring 3.0 - AOP - Which of the following execution pointcut expressions will NOT map to the doStuff() method below?

```
package com.skillguru.spring;

public class Foo implements DoAble {

    public void doStuff() {

    }

}
```

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	execution(public **(..))	INCORRECT: This maps to public methods with any return types, any methods, any args. It will therefore map to the doStuff() method defined.
	<input type="radio"/>	execution(* com.skillguru..*.*(..))	INCORRECT: This maps to any return types, to any types in the defined package or any sub-package, any methods, any args. It will therefore map to the doStuff() method defined.
	<input type="radio"/>	execution(* do*(..))	INCORRECT: This maps to any return types, to any types, any methods beginning with 'do', with no parameters. It will therefore map to the doStuff() method defined.
✓	<input type="radio"/>	execution(* com.skillguru.spring.DoAble.*(..))	The last asterisk means that it will match a method with a single argument. doStuff() does not accept any parameters. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/aop.html#aop-pointcuts-examples Ref: http://www.eclipse.org/aspectj/doc/released/progguide/semantics-pointcuts.html

Question :27

Spring 3.0 - Remoting - Spring provides support for which of the following standard Java web service APIs?

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	Exposing web services using JAX-RPC	
	<input type="radio"/>	Accessing web services using JAX-RPC	
	<input type="radio"/>	Exposing web services using JAX-WS	
	<input type="radio"/>	Accessing web services using JAX-WS	
✓	<input type="radio"/>	All of the above	All of these were supported from Spring 2.5.6 onwards. JAX-RPC 1.1 is the legacy web service API in J2EE 1.4. JAX-RPC has largely been superseded by the annotation based JAX-WS 2.x. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/remoting.html#remoting-web-services

Question :28

Spring 3.0 - Web & MVC - Which of the following statements regarding Annotation-based controller configuration is TRUE?

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	@Mapping allows Controller classes to be mapped to specific URIs and methods to be mapped to either GET or POST http requests.	INCORRECT: There is no @Mapping annotation The @RequestMapping annotation is used to map URLs like '/edit.do' onto a class or a particular handler method Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/mvc.html#mvc-ann-requestmapping
✓	<input checked="" type="radio"/>	To enable autodetection of annotated controllers, you have to add component scanning to your configuration.	CORRECT: It is necessary to enable component scanning. This is Spring's cue to read the underlying metadata and process it accordingly. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/mvc.html#mvc-ann-sessionattrib
	<input type="radio"/>	@Controller annotation indicates that a particular class serves the role of a controller. As well as annotating it is also necessary to extend one of the standard controller base classes.	INCORRECT: It is not necessary to extend a controller class. It is sufficient to just annotate with @Controller. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/mvc.html#mvc-ann-sessionattrib
	<input type="radio"/>	@SessionParams annotation declares session attributes used by a specific handler.	INCORRECT: There is no @SessionParams annotation The @SessionAttributes annotation is used for this purpose. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/mvc.html#mvc-ann-sessionattrib

Question :29

Spring 3.0 - Container & Test - Which if the following is NOT true regarding the @Autowired and @Required annotations?

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	Use of the annotations requires that certain BeanPostProcessors are first registered within the Spring container.	INCORRECT: This statement is TRUE. The questions requires you to select the FALSE option. You need to 'switch on' annotations in Spring. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-annotation-config
	<input type="radio"/>	These annotations may be individually and explicitly registered in the container using RequiredAnnotationBeanPostProcessor and AutowiredAnnotationBeanPostProcessor .	INCORRECT: This statement is TRUE. The questions requires you to select the FALSE option. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-annotation-config
	<input type="radio"/>	These annotations may be implicitly registered in the container using <context:annotation-config/> from the context namespace.	INCORRECT: This statement is TRUE. The questions requires you to select the FALSE option. Note that the implicitly registered post-processors include AutowiredAnnotationBeanPostProcessor, CommonAnnotationBeanPostProcessor, PersistenceAnnotationBeanPostProcessor, and RequiredAnnotationBeanPostProcessor. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-annotation-config
✓	<input type="radio"/>	It is sufficient to configure with just annotations in the source code without any additional configuration in the Spring container.	CORRECT: This statement is NOT true. If you annotate code without also 'switching on' the annotations in the configuration they will have no effect and likely result in exceptions later on as properties are left uninitialized. There are two approaches to 'switch on' annotations - these are described in the answers above. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-annotation-config

Question :30

Spring 3.0 - Container & Test - Which of the following are JSR-250 annotation/s?

Correct Answer	Your Selection	Answer	Explanation
✓	<input type="checkbox"/>	@Resource	CORRECT: This is a JSR-250 annotation. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-resource-annotation
✓	<input type="checkbox"/>	@PostConstruct	CORRECT: This is a JSR-250 annotation. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-postconstruct-and-predestroy-annotations
	<input type="checkbox"/>	@Autowired	INCORRECT: This is NOT a JSR-250 annotation; it is a Spring annotation. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-annotation-config Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org.springframework.beans.factory.annotation/Autowired.html
✓	<input type="checkbox"/>	@PreDestroy	CORRECT: This is a JSR-250 annotation. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-postconstruct-and-predestroy-annotations

Question :31

Spring 3.0 - Web & MVC - Controllers, ViewResolvers and HandlerMappings are key components of Spring MVC. Which of the following 2 statements about these components are TRUE?

Correct Answer	Your Selection	Answer	Explanation
✓	<input type="checkbox"/>	Controllers are the components that form the 'C' part of the MVC.	CORRECT: MVC stands for Model View and Controller
	<input type="checkbox"/>	Handler mappings are components capable of resolving view names to views.	INCORRECT: A HandlerMapping maps URL Requests to controllers. The given description explains the behaviour of 'ViewResolvers'.
✓	<input type="checkbox"/>	AbstractWizardFormController is deprecated as of Spring 3.x in favour of annotated controllers	CORRECT: Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org.springframework.web.portlet.mvc/AbstractWizardFormController.html
	<input type="checkbox"/>	UrlBasedViewResolver is a HandlerMapping that maps symbolic view names to URLs, without an explicit mapping definition.	INCORRECT: UrlBasedViewResolver is a type of ViewResolver, and not a HandlerMapping. A ViewResolver is the component that deals with the 'view' part of the 'ModelAndView' returned from a controller. It ultimately determines what the user will see in the response. A HandlerMapping maps URL Requests to controllers.

Question :32

Spring 3.0 - JDBC - Which of the following is NOT TRUE of the JdbcTemplate class?

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	JdbcTemplate handles the creation and release of resources.	INCORRECT: This statement is TRUE. The question required you to select the statement which is NOT true.
	<input type="radio"/>	JdbcTemplate executes statement creation and execution.	INCORRECT: This statement is TRUE. The question required you to select the statement which is NOT true.
✓	<input type="radio"/>	JdbcTemplate provides default caching of results limiting throughput to the underlying datasource.	CORRECT: This statement is NOT true. The developer will need to configure a caching provider such as ehcache. Caching is not a default feature of JdbcTemplate although there is extensive support in Spring for this technology. Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org.springframework.jdbc.core/JdbcTemplate.html
	<input type="radio"/>	JdbcTemplate catches JDBC exceptions and translates them to the generic persistence-technology-independent exceptions.	INCORRECT: This statement is TRUE. The question required you to select the statement which is NOT true.

Question :33

Spring 3.0 - Container & Test - Which of the following statements regarding Spring Integration Testing is FALSE ?

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	AbstractTransactionalDataSourceSpringContextTests is deprecated in Spring 3.0.	INCORRECT: This statement is TRUE. The question required you to select the FALSE statement. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/testing.html#integration-testing-overview
✓	<input type="radio"/>	TestContext defines a listener API for reacting to test execution events	CORRECT: This statement is FALSE. TestContext definition: "Encapsulates the context in which a test is executed, agnostic of the actual testing framework in use." The description provided is the given definition of the "TestExecutionListener" Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/testing.html#testcontext-key-abstractions
	<input type="radio"/>	The annotation @DirtiesContext should be used if you wish to force the applicationContext to close after the execution of a test-class or method.	INCORRECT: This statement is TRUE. The question required you to select the FALSE statement. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/testing.html#integration-testing-annotations
	<input type="radio"/>	The annotation @ContextConfiguration can be used at the class level to declare how to load and configure an ApplicationContext.	INCORRECT: This statement is TRUE. The question required you to select the FALSE statement. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/testing.html#integration-testing-annotations

Question :34

Spring 3.0 - Container & Test - How are dependencies injected in to an object?

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	Through constructor arguments	
	<input type="radio"/>	Arguments to a factory method	
	<input type="radio"/>	Properties which are set on the object instance after it has been constructed or returned from a factory method	
✓	<input type="radio"/>	All of the above	Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-dependencies

Question :35

Spring 3.0 - Container & Test - Which of the below describe a benefit of dependency injection?

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	It can conveniently externalize dependencies in to a configuration file/s	
	<input type="radio"/>	It promotes easier unit testing as it allows swapping out of dependent classes for mock, or stubbed, objects	
	<input type="radio"/>	It promotes greater decoupling between application classes	
✓	<input type="radio"/>	All of the above	Although an external configuration file/s are not the only option for configuring dependencies they do allow dependencies between collaborators to be abstracted from the code itself and in to a file. This makes the first answer correct. Because of this it is possible to define additional beans to supplement a unit-testing environment outside of a container whereby dependent objects, such as a DAO, may be swapped out for a mocked or stubbed equivalent that does not actually connect to the underlying datasource or other dependencies. This makes the second correct. All of this is made possible because the classes themselves are not strongly coupled to each other. So the third answer is also correct.

Question :36

Spring 3.0 - SpEL - Which of the following two statements are TRUE regarding the Spring Expression Language

Correct Answer	Your Selection	Answer	Explanation
✓	<input type="checkbox"/>	Expression.getValue(Class<T> desiredResultType) with throw an EvaluationException if the value cannot be cast to the type T.	CORRECT: Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/expressions.html#expressions-evaluation
✓	<input type="checkbox"/>	The interface ExpressionParser is responsible for parsing an expression string.	CORRECT: Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/expressions.html#expressions-evaluation
	<input type="checkbox"/>	Expression.getValue(Class<T> desiredResultType) with throw a ParseException if the value cannot be cast to the type T.	INCORRECT: See the first answer for the correct exception signature of this method. The ParseException is actually thrown by the ExpressionParser.parseException() method. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/expressions.html#expressions-evaluation
	<input type="checkbox"/>	The interface ExpressionParser is responsible for evaluating the previously defined expression string.	INCORRECT: See the second answer for the correct definition of the ExpressionParser. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/expressions.html#expressions-evaluation

Question :37

Spring 3.0 - JMS - Which of the following statements about Spring's JMS support is FALSE ?

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	Spring translates the checked exceptions from the javax.jms package in to unchecked ones.	INCORRECT: This statement is TRUE. The question required you to select the FALSE statement.
	<input type="radio"/>	JmsTemplate assists in "boiler plate" code such as creating connections, obtaining sessions, and the sending and/or receiving of messages.	INCORRECT: This statement is TRUE. The question required you to select the FALSE statement.
✓	<input type="radio"/>	Spring's support for both JMS 1.0.2 and 1.1+ is conveniently abstracted away behind the single JmsTemplate class.	CORRECT: This statement is FALSE. There are two options for the functionality offered by the JmsTemplate: the JmsTemplate uses the JMS 1.1 API, and the (now deprecated) subclass JmsTemplate102 uses the JMS 1.0.2 API. Ref: http://static.springsource.org/spring/docs/2.5.x/reference/jms.html#jms-using
	<input type="radio"/>	In order to perform the work of translating a javax.jms.Message in to a POJO (in your domain model) an implementation of MessageConverter can be written which is then injected in to the JmsTemplate.	INCORRECT: This statement is TRUE. The question required you to select the FALSE statement.

Question :38

Spring 3.0 - Web & MVC - Which of the following are valid return types, or values, from a Controller implementation?

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	null	The ModelAndView returned from a Controller may be null (if the controller itself takes responsibility for the view).
	<input type="radio"/>	A String view name	The ModelAndView returned from a Controller can contain a simple String - which the configured ViewResolver must map to a view.
	<input type="radio"/>	A String view name with a Map of model objects.	The ModelAndView returned from a Controller can contain a Map of model objects which may be used to display data back to the user via the configured view.
✓	<input type="radio"/>	All of the above	CORRECT: Because of above explanations Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org.springframework.web.servlet.mvc/Controller.html#handleRequest%28javax.servlet.http.HttpServletRequest,%20javax.servlet.http.HttpServletResponse%29 Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org.springframework.web.portlet/ModelAndView.html

Question :39

Spring 3.0 - Remoting - Which of the following TWO statements are TRUE about ServletEndpointSupport?

Correct Answer	Your Selection	Answer	Explanation
	<input type="checkbox"/>	Using the ServletEndpointSupport, it is possible to expose a Java interface as a RESTful service.	FALSE This class does not provide these feature.
✓	<input type="checkbox"/>	It is used to expose a servlet based web service using JAX-RPC.	TRUE: ServletEndpointSupport is used to expose a servlet based web service using JAX-RPC. Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/remoting/jaxrpc/ServletEndpointSupport.html
	<input type="checkbox"/>	This class is a simple exporter for JAX-WS services, autodetecting annotated service beans (through the JAX-WS WebService annotation).	FALSE: This describes SimpleHttpServerJaxWsServiceExporter in the org.springframework.remoting.jaxws package that has superseded this legacy class. Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/remoting/jaxws/SimpleJaxWsServiceExporter.html
✓	<input type="checkbox"/>	This class is deprecated as of Spring 3.0 and the JAX-WS APIs are recommended as it's replacement.	TRUE: See reference below Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/remoting/jaxrpc/ServletEndpointSupport.html

Question :40

Spring 3.0 - Web & MVC - Which of the following view technologies are supported by Spring?

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	JSTL/JSP	
	<input type="radio"/>	MS Excel	
	<input type="radio"/>	Adobe PDF	
✓	<input type="radio"/>	All of the above	<p>Spring provides support for all of the above views. For JSP or JSTL is achieved using a view resolver defined in the <code>WebApplicationContext</code>. For Excel, there are two choices: either subclass <code>org.springframework.web.servlet.view.document.AbstractExcelView</code> (for Excel files generated by POI) or <code>org.springframework.web.servlet.view.document.AbstractJExcelView</code> (for JExcelApi-generated Excel files). These abstract classes will require the developer to implement the <code>buildExcelDocument()</code> method. For PDF subclass <code>org.springframework.web.servlet.view.document.AbstractPdfView</code> and implement <code>buildPdfDocument()</code>. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/view.html#view-jsp Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/view.html#view-document</p>

Question :41

Spring 3.0 - Remoting - Which of the following TWO statements are TRUE about ServletEndpointSupport?

Correct Answer	Your Selection	Answer	Explanation
✓	<input type="checkbox"/>	It is used to expose a servlet based web service using JAX-RPC.	TRUE: ServletEndpointSupport is used to expose a servlet based web service using JAX-RPC. Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org.springframework.remoting.jaxrpc/ServletEndpointSupport.html
✓	<input type="checkbox"/>	It is a convenience base class for JAX-RPC servlet endpoint implementations.	TRUE: See references above
	<input type="checkbox"/>	This is a FactoryBean for a specific port of a JAX-RPC service.	FALSE: This is a description of JaxRpcPortProxyFactoryBean which is used to access web services using JAX-RPC Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org.springframework.remoting.jaxrpc/JaxRpcPortProxyFactoryBean.html
	<input type="checkbox"/>	This class exposes a proxy for the port, to be used for bean references.	FALSE: This is a description of the now deprecated JaxRpcPortProxyFactoryBean which is used to access web services using JAX-RPC. JAX-WS is now the preferred approach. Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org.springframework.remoting.jaxrpc/JaxRpcPortProxyFactoryBean.html

Question :42

Spring 3.0 - Remoting - You have been tasked for creating a remote service and client using Java and Spring. No amount of bribing Dave, the networks guy, with coffee will convince him to open up alternative ports in the network's firewalls other than the standard 80 and 443.

Which would NOT be a suitable technology from the list below?

Correct Answer	Your Selection	Answer	Explanation
✓	<input checked="" type="radio"/>	RMI	CORRECT: RMI is NOT a suitable technology choice in this scenario. Port 80 and 443 are the standard ports for http and https this limits you to not using the RMI protocol. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/remoting.html#remoting-considerations
	<input type="radio"/>	HttpInvoker	INCORRECT: This would be a valid technology choice as it will use port 80. The client is Java.
	<input type="radio"/>	Web service	INCORRECT: This would be a valid technology choice as it will use port 80. A Java client could be used (although it could be any technology).
	<input type="radio"/>	Hessian or Burlap	INCORRECT: This would be a valid technology choice as it will use port 80. A Java client could be used.

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	Like the JdbcTemplate, an instance of JmsTemplate is thread-safe once configured. It is therefore safe to configure one instance and share this between beans.	INCORRECT: This statement is true. The question required you to identify the false statement. JmsTemplate is thread-safe and is therefore suitable to share between collaborators.
	<input type="radio"/>	JmsTemplate's send operations are compatible with both Queues and Topics.	INCORRECT: This statement is true. The question required you to identify the false statement. JmsTemplate is compatible with both modes of JMS, i.e. queues and topics.
✓	<input checked="" type="radio"/>	JMS is principally concerned with asynchronous processing. Because of this the receive operations of the JmsTemplate are asynchronous .	CORRECT: This statement is false. JmsTemplate receive() methods are blocking / synchronous in nature. When using these methods it is recommended to set the timeout property so they do not block indefinitely. For asynchronous consumption it is recommended to implement the MessageListener interface and configure a MessageListenerContainer injecting your MessageListener. JmsTemplate API - Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/jms/core/JmsTemplate.html Asynchronous processing via listeners ... Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/classic-spring.html#d0e43036 Receive methods - Ref: http://static.springsource.org/spring/docs/2.5.x/reference/jms.html#jms-receiving-sync
	<input type="radio"/>	JmsTemplate provides support for conversion of both outgoing and incoming messages to your underlying data model with support from the MessageConverter interface.	INCORRECT: This statement is true. The question required you to identify the false statement. MessageConverters - http://static.springsource.org/spring/docs/2.5.x/reference/jms.html#jms-msg-conversion JmsTemplate provides various convertAndSend() and receiveAndConvert() methods to convert messages from and to your domain model.

Question :44

Spring 3.0 - JMX - Which two steps are valid when using annotations to declare managed / MBeans?

Correct Answer	Your Selection	Answer	Explanation
	<input type="checkbox"/>	Annotate the class with @MBeanResource and the operation/s with @MBeanOperation	INCORRECT: These annotations do not exist.
✓	<input type="checkbox"/>	Annotate the class with @ManagedResource and the operation/s with @ManagedOperation	CORRECT: The annotations @ManagedResource and @ManagedOperation are the correct ones to use to declare/export a bean and its methods as "managed". Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jmx.html#jmx-interface-metadata
✓	<input type="checkbox"/>	Declare an assembler of type MetadataMBeanInfoAssembler	CORRECT: The assembler MetadataMBeanInfoAssembler is the correct assembler and approach. This is injected in to an MBeanExporter. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/jmx.html#jmx-interface-metadata
	<input type="checkbox"/>	Declare an exporter of type MetadataMBeanExporter	INCORRECT: There is no such class as MetadataMBeanExporter.

Question :45

Spring 3.0 - Container & Test - Which of the following statements is FALSE regarding bean naming?

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	A bean may be defined without an id or a name.	INCORRECT: This statement is true. The question required you to select the FALSE option. Anonymous beans do not require an id or a name and are legal in Spring.
	<input type="radio"/>	It is possible for a bean to have multiple names.	INCORRECT: This statement is true. The question required you to select the FALSE option. Multiple bean names may be defined - they can be separated by a comma (,), semicolon(;), or whitespace in the 'name' attribute.
✓	<input type="radio"/>	The following two declarations are both legal... <bean id="/myBean" class="foo.Bar" /> ...or... <bean name="/myBean" class="foo.Bar" />	CORRECT: This statement is FALSE. The ID attribute is a real XML element and, as such, the XML specification limits the characters that are legal in XML ids. '/' is not allowed in an id although it is fine in the "name" attribute. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.htm#beans-beannames
	<input type="radio"/>	A bean may be given multiple names via the <alias> tag	INCORRECT: This statement is true. The question required you to select the FALSE option. Aliases may be used as a means of applying multiple names to a bean.

Question :46

Spring 3.0 - Container & Test - Which of the following are legal values when using the 'dependency-check' attribute in the bean definition?

Correct Answer	Your Selection	Answer	Explanation
✓	<input checked="" type="radio"/>	none, simple, object, all	CORRECT: These are the valid options for this attribute. This attribute allows properties which are not exposed via usual means (setters, constructor, factory method) to be enforced. Dependency checking modes: "none" - No dependency checking. Properties of the bean which have no value specified for them are simply not set. "simple"- Dependency checking is performed for primitive types and collections (everything except collaborators). "object" - Dependency checking is performed for collaborators only. "all" - Dependency checking is done for collaborators, primitive types and collections. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-factory-dependencies
	<input type="radio"/>	true/false	INCORRECT: These are not valid options for the 'dependency-check' attribute.
	<input type="radio"/>	byName, byType, constructor	INCORRECT: These are not valid options for the 'dependency-check' attribute. These are in fact the options valid for the 'autowire' attribute.
	<input type="radio"/>	None of these. Dependencies are only managed by constructors, setters or factory methods.	INCORRECT: See correct answer for the actual options.

Question :47

Spring 3.0 - Transactions - Which best describes the propagation attribute REQUIRES_NEW?

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	Support a current transaction, create a new one if none exists.	INCORRECT: This is a description of the REQUIRED propagation attribute.
✓	<input checked="" type="radio"/>	Create a new transaction, suspend the current transaction if one exists.	CORRECT: This is a correct description of the REQUIRES_NEW propagation attribute. Ref: http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/transaction/annotation/Propagation.html
	<input type="radio"/>	Support a current transaction, throw an exception if none exists.	INCORRECT: This is a description of the MANDATORY propagation attribute.
	<input type="radio"/>	Execute non-transactionally, suspend the current transaction if one exists.	INCORRECT: This is a description of the NOT_SUPPORTED propagation attribute.

Question :48

Spring 3.0 - Container & Test - From the numbered list of lifecycle events below, which of the following options indicates the order of execution?

1. Set properties of the bean
2. Construction of the bean
3. An implementation of postProcessBeforeInitialization() from the BeanPostProcessor interface (assume the implementing bean is correctly registered).
4. init-method

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	2, 3, 4, 1	INCORRECT: See correct answer for explanation of order.
✓	<input checked="" type="radio"/>	2, 1, 3, 4	CORRECT: From the given options, Spring will execute the lifecycle events in the following order: First it will construct the bean. Second, it will call any setters on the bean as defined in the configuration. The BeanPostProcessor potentially executes before and/or after "init-method". In this example the "postProcessBeforeInitialization()" method is specifically mentioned. In which case this method executes next and before the "init-method" which executes last. This reference details how an Initialization callbacks via an init-method occurs after the properties are set. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-factory-lifecycle-initializingbean The following reference details how the BeanPostProcessor contains two methods one that is executed before any initialization callback and one that is executed after. The one detailed in this question occurs before. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/beans.html#beans-factory-extension-bpp
	<input type="radio"/>	3, 2, 4, 1	INCORRECT: See correct answer for explanation of order.
	<input type="radio"/>	3, 2, 1, 4	INCORRECT: See correct answer for explanation of order.

Question :49

Spring 3.0 - AOP - What is a pointcut?

Correct Answer	Your Selection	Answer	Explanation
	<input type="radio"/>	A Java class encapsulating one or more advices	INCORRECT: This describes an Aspect.
	<input type="radio"/>	A point during the execution of a program, such as the execution of a method or the handling of an exception. In Spring AOP, this always represents a method execution.	INCORRECT: This describes a Join Point
✓	<input type="radio"/>	It determines join points of interest, and thus enable us to control when advice executes.	CORRECT: This is a correct definition of a pointcut. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/aop.html#aop-introduction-defn
	<input type="radio"/>	Action taken by an aspect at a particular join point. Different types include "around," "before" and "after".	INCORRECT: This describes an Advice.

Question :50

Spring 3.0 - Web & MVC - Which of the following 2 statements regarding Annotation-based controller configuration is TRUE?

Correct Answer	Your Selection	Answer	Explanation
✓	<input type="checkbox"/>	The @CookieValue annotation allows a method parameter to be bound to the value of an HTTP cookie.	CORRECT Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/mvc.html#mvc-ann-cookievalue
	<input type="checkbox"/>	The @ResponseBody annotation can be put on a method to indicate that the return type should be placed in a Model, or interpreted as a view name.	INCORRECT: The @ResponseBody annotation is similar to @RequestBody. This annotation can be put on a method and indicates that the return type should be written straight to the HTTP response body (and not placed in a Model, or interpreted as a view name). Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/mvc.html#mvc-ann-responsebody
	<input type="checkbox"/>	@Mapping allows Controller classes to be mapped to specific URIs and methods to be mapped to either GET or POST http requests.	INCORRECT: There is no @Mapping annotation The @RequestMapping annotation is used to map URLs like '/edit.do' onto a class or a particular handler method Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/mvc.html#mvc-ann-requestmapping
✓	<input type="checkbox"/>	To enable autodetection of annotated controllers, you have to add component scanning to your configuration.	CORRECT: It is necessary to enable component scanning. This is Spring's cue to read the underlying metadata and process it accordingly. Ref: http://static.springsource.org/spring/docs/3.0.x/spring-framework-reference/html/mvc.html#mvc-ann-sessionattrib