javaTpoint Search on javatpoint...

Try Another Quiz

Question: 1 What is/are typically case(s) where you usually need to manually instanciated an ApplicationContext?

Your Answer: In a standalone application started with a main method ✓

Correct Answer: In a standalone application started with a main method

Description: 1. In a web application, the ContextLoaderListener is in charge to create an WebApplicationContext.
2. In an integration test based on Spring, the SpringJUnit4ClassRunner creates the application context for you. The @ContextConfiguration annotation allows to specified application context configuration files.
3. In a main method, you have to instanciated a class implementing the ApplicationContext interface (examples: ClassPathXmlApplicationContext or FileSystemXmlApplicationContext)

Question: 2 Given the following Spring configuration file, what is the correct answer:
```
<bean class="com.spring.service.MyServiceImpl">
<property name="repository" ref="jpaDao"/>
</bean>
<bean class="com.spring.repository.JpaDao"/>
```

Your Answer: Answers 1 and 2 are both rights ✗

Correct Answer: The second declared bean JpaDao is missing an id must be named jpaDao

Description: Those beans are anonymous because no id is supplied explicitly. Thus Spring container generates a unique id for that bean. It uses the fully qualified class name and appends a number to them. However, if you want to refer to that bean by name, through the use of the ref element you must provide a name. To be correct, the 2nd bean has to declare a jpaDao id attribute in order to be reference by the repository property of the first bean.

Question: 3 How could you externalize constants from a Spring configuration file or a Spring annotation into a .properties file? Select one or more answers

Your Answer: By using the context:property-placeholder tag ✓

Correct Answer: By using the context:property-placeholder tag

Description: 1. The <util:constant static-field="constant name"/> tag enables to reference a Java constant or enumeration into a spring configuration file
2. ConstantPlaceholderConfigurer does not exist. You may think about the PropertyPlaceholderConfigurer bean post processor.
3. The <context:property-placeholder location="file:/myApp.properties" /> tag activates the replacement of ${...} placeholders, resolved against the specified properties file.
4. The c: namespace is for simplifying constructor syntax (since Spring 3.1) and don´t provide such feature.

Question: 4 How is named the bean that is defined in the following configuration class. Select a single answer.
```
@Configuration
public class ApplicationConfig {
@Autowired
private DataSource dataSource;
@Bean
ClientRepository clientRepository() {
ClientRepository accountRepository = new JpaClientRepository();
accountRepository.setDataSource(dataSource);
return accountRepository;
}
}
```

Your Answer: Two beans are defined : a data souce and a repository ✗

Correct Answer: clientRepository

Description: The @Bean annotation defines a String bean with the id clientRepository. JpaClientRepository is the implementation class of the bean. The data source is injected and is not declared in this class.

Question: 5 What are the main advantages of using interfaces when designing business services? Select answer.

Your Answer: Mocking or stubbing the service ✓

Correct Answer: Mocking or stubbing the service

Description: 1. With modern mock API like Mockito or EasyMock, interfaces are not mandatory for mocking or stubbing the service. But using interface remains easier when you have to manually mock the service in unit test.
2. Auto-injection is possible with class. Spring uses CGLIB.
3. Dependency checking is an advantage of dependencies injection.

Question: 6 What one is not the right affirmations about the @PostConstruct, @Resource and the @PreDestroy annotations?

Your Answer: The context:annotation-config tag enable them ✗

Correct Answer: The Spring Framework embedded those annotation

Description: None

Question: 7 Given the Spring configuration file, which are the correct statements?
<bean class="com.spring.service.BankServiceImpl"
p:bankName="NationalBank">
</bean>

Your Answer: Bean id is bankServiceImpl ✗

Correct Answer: The p namespace has to be declared

Description: None

Question: 8 How to auto-inject into a field a bean by its name? Select one or more response.

Your Answer: With the name attribute of the @Autowired annotation ✗

Correct Answer: By using both the @Autowired and the @Qualifier spring annotations

Description: None

Question: 9 Select the right statement about referring a Spring configuration file inside the package com.example.myapp in the below example?
ApplicationContext context = new
ClassPathXmlApplicationContext("classpath:/com.example.myapp.config.xml");

Your Answer: All of the above ✓

Correct Answer: All of the above

Description: 1. When using the ClassPathXmlApplicationContext, the classpath: prefix is default one so you could omit it
2. In a Spring location resource, package separator is a slash and not a dot. Thus the com/example/myapp/config.xml syntax has to be used.
3. ClassPathXmlApplicationContext starts looking from root of the classpath regardless of whether specify "/"

Question: 10 What statement is not correct in live environment? Select a unique answer.

Your Answer: All of the above ✓

Correct Answer: All of the above

Description: 1. You may auto-wiring properties by constructor, setter or properties in the same bean
2. The <constructor-arg> tag helps to instanciated a bean without default or no-args constructor
3. The <constructor-arg> tag could take type and index to reduce ambiguity, but not name which requires debug symbols.

Finish

Tweet Like 127K G+