1. For building RESTful Web Services in Spring Boot, what is the required dependency?

A: spring-boot-starter-web

B: spring-boot-starter-webmvc

C: spring-boot-starter-rest

D: spring-boot-html

In Spring AOP, \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ is responsible for choosing whether to proceed to the join point or to shortcut the advised method execution by returning its own return value or throwing an exception

A: MethodInterceptor

B: MethodBeforeAdvice

C: ThrowsAdvice

D: MethodAroundAdvice

**Which Spring annotation is used to create RESTful web services using Spring MVC?**

1. @RestController
2. @Controller
3. @Component
4. @Rest

**@RestController annotation is a combination of the below two annotations**

1. @Component and @ResponseBody annotations
2. @Controller and @ResponseBody annotations
3. @Service and @ResponseBody annotations
4. None of the above

**Which Spring annotation is used to handle HTTP POST requests?**

1. @GetMapping
2. @PutMapping
3. @CreateMapping
4. @PostMapping

**What is the purpose of the Spring Boot DevTools?**

1. To provide a way to run Spring Boot applications in a development environment
2. To provide hot-reloading of code changes and automatic application restart
3. To provide a way to run Spring Boot applications as a service
4. To provide a way to access the underlying database of a Spring Boot application

**What is the difference between Spring and Spring Boot?**

1. Spring is a Java framework, while Spring Boot is a Java library
2. Spring Boot is an opinionated version of Spring, providing a set of default configurations
3. Spring Boot is a lightweight version of Spring, while Spring is a full-featured framework
4. Spring Boot is a front-end framework, while Spring is a back-end framework

**What are the Spring annotations used for Exception handling?**

1. @ControllerAdvice
2. @ExceptionHandler
3. @ResponseStatus
4. All of the above

**How can you access request parameters in a Spring MVC controller method?**

a) By using the @RequestParams annotation.

b) By accessing them directly from the HttpServletRequest object.

c) By using the @PathVariable annotation.

d) By using the @RequestParam annotation.

e) By using the @ReqParameter annotation.

**What is the aim of Spring Data JPA?**

a) Spring Data JPA aims to reduce the amount of boilerplate code required for common database operations.

b) Spring Data JPA aims to provide the implementation for JPA interfaces

c) Spring Data JPA is a JPA provider and implementation for JPA interfaces

d) Spring Data JPA is an ORM framework that provides an implementation for JPA interfaces

**Which annotation marks a class as an entity in JPA?**

a) @EntityClass

b) @JPAEntity

c) @Entity

d) @TableEntity

Which annotation is used to enable Spring’s aspect-oriented programming (AOP) support?

**(a) @Aspect**

**(b) @AspectJ**

**(c) @EnableAop**

**(d) @EnableAspectJAutoProxy**

What is the purpose of the @Qualifier annotation in Spring?

**(a) It specifies the order of bean initialization among multiple beans**

**(b) It marks a bean for automatic injection**

**(c) It resolves ambiguity when multiple beans of the same type are present**

**(d) It marks a class to become a Spring bean**

Which of the following tag is required to enable component scanning in Spring XML configuration?

**(a) <context:scan package="com.example" />**

**(b) <context:scan-components base-package="com.example" />**

**(c) <context:component-scan base-package="com.example" />**

**(d) <scan: component base-package="com.example" />**

To access a bean property, which SpEL expression is correct?

**(a) #{beanName.propertyName}**

**(b) ${beanName.propertyName}**

**(c) @beanName.propertyName**

**(d) %beanName.propertyName%**

Which annotation is used to mark a method as an exception handling method in Spring MVC?

**(a) @ExceptionHandler**

**(b) @ExceptionHandling**

**(c) @HandleException**

**(d) @ExceptionResolver**

If you want to write global exception handling code that can be applied to multiple controllers in Spring MVC, which annotation will you use?

**(a) @GlobalExceptionHandling**

**(b) @GlobalExceptionHandler**

**(c) @ControllerAdvice**

**(d) @EnableGlobalExceptionHandling**

In Spring JDBC, which method will you use to execute a SQL query that can map the results to a Java object?

**(a) executeQueryForObject()**

**(b) fetchObject()**

**(c) executeForObject()**

**(d) queryForObject()**

What is the role of the DispatcherServlet in Spring MVC?

**(a) It configures the Spring MVC dependencies**

**(b) It manages the application's database connections**

**(c) It handles incoming requests and delegates them to controllers**

**(d) It handles security authentication and authorization.**

Which pointcut expression is correct for the below statement?

“Any method with zero or more parameters inside EmpDao class”

**(a) public \* com.dev.dao.EmpDao.\*( )**

**(b) public \* com.dev.dao.\*.\*(..)**

**(c) public \* com.dev.dao.EmpDao.\*(..)**

**(d) public Integer com.dev.dao.EmpDao.\*( )**

In the context of Spring AOP, select the method which is the candidate for below Pointcut Expression?

**public \* \*(Invoice)**

**(a) public Integer saveInvoice(Integer id){….}**

**(b) public void deleteInvoice(Integer id){….}**

**(c) public void updateInvoice(Invoice inv){….}**

**(d) public Invoice getInvoice(Integer id){….}**

**What will be the result of the following code snippet when doWork() is called?**

**@Component**

**public class MyService {**

**@Autowired**

**private MyRepository repository;**

**public void doWork() {**

**System.***out***.println("Repository: " + repository);**

**}**

**}**

**@Repository**

**public class MyRepository {**

**}**

Assuming the Spring configuration is properly set up to scan the package:

<context:component-scan base-package=”com.example”/>

A) Repository: null  
B) Repository: com.example.MyRepository@someHashCode  
C) Compilation error  
D) Runtime error

**What does the following Spring configuration achieve?**

**@Configuration**

**public class AppConfig {**

**@Bean**

**public MyBean myBean() {**

**return new MyBean();**

**}**

**}**

A) Declares a MyBean instance as a Spring bean  
B) Declares MyBean as a singleton bean  
C) Declares MyBean as a prototype bean  
D) Both A and B

**Given the following code, what will be the output when the method printProperty() is called?**

**@Component**

**public class MyComponent {**

**@Value("${my.property}")**

**private String myProperty;**

**public void printProperty() {**

**System.*out*.println("Property: " + myProperty);**

**}**

**}**

Assuming the property is set in an application.properties file:

**my.property=HelloSpring**

A) Property: null  
B) Property: HelloSpring  
C) Property: ${my.property}  
D) Runtime error

**Which annotation is used to define a bean’s scope as prototype in Spring?**

A) @Component (“prototype”)  
B) @Scope(“prototype”)  
C) @Prototype  
D) @PrototypeScope

**What will happen if you run the following Spring configuration?**

**<bean id="myService" class="com.example.MyService" init-method="init" destroy-method="cleanup">**

**<property name="dependency" ref="myDependency"/>**

**</bean>**

**<bean id="myDependency" class="com.example.MyDependency"/>**

Assuming MyService class is defined as:

**public class MyService {**

**private MyDependency dependency;**

**public void setDependency(MyDependency dependency) {**

**this.dependency = dependency;**

**}**

**public void init() {**

**System.*out*.println("MyService initialized");**

**}**

**public void cleanup() {**

**System.*out*.println("MyService cleaned up");**

**}**

**}**

A) MyService initialized  
B) MyService cleaned up  
C) Both A and B  
D) None of the above

**How can you mark a field as the primary key in a JPA entity?**

a) @PrimaryKey

b) @EntityKey

c) @Id

d) @Key

**How can you customize the column name for a field in a JPA entity?**

a) @Column(name="custom\_name")

b) @FieldName("custom\_name")

c) @TableColumn("custom\_name")

d) @DatabaseColumn("custom\_name")