Spring Framework – Java Interview Questions

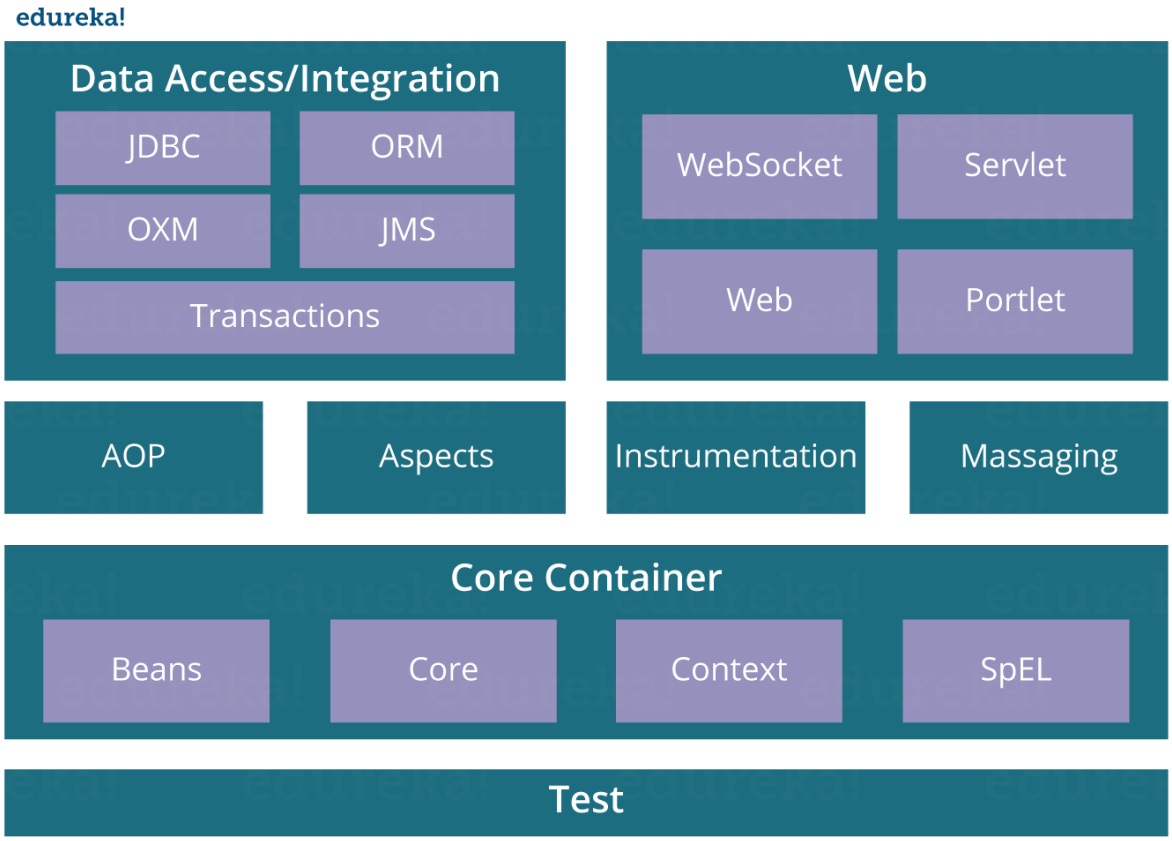
Q1. What is Spring?

Wikipedia defines the Spring framework as “an application framework and inversion of control container for the Java platform. The framework’s core features can be used by any Java application, but there are extensions for building web applications on top of the Java EE platform.” Spring is essentially a lightweight, integrated framework that can be used for developing enterprise applications in java.

Q2. Name the different modules of the Spring framework.

Some of the important Spring Framework modules are:

- Spring Context – for dependency injection.
- Spring AOP – for aspect oriented programming.
- Spring DAO – for database operations using DAO pattern
- Spring JDBC – for JDBC and DataSource support.
- Spring ORM – for ORM tools support such as Hibernate
- Spring Web Module – for creating web applications.
- Spring MVC – Model-View-Controller implementation for creating web applications, web services etc.



Q3. List some of the important annotations in annotation-based Spring configuration.

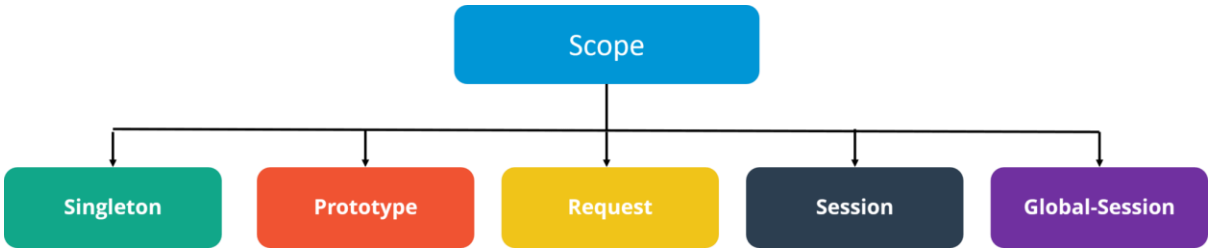
The important annotations are:

- @Required
- @Autowired
- @Qualifier
- @Resource
- @PostConstruct
- @PreDestroy

Q4. Explain Bean in Spring and List the different Scopes of Spring bean.

Beans are objects that form the backbone of a Spring application. They are managed by the Spring IoC container. In other words, a bean is an object that is instantiated, assembled, and managed by a Spring IoC container.

There are five Scopes defined in Spring beans.



- **Singleton**: Only one instance of the bean will be created for each container. This is the default scope for the spring beans. While using this scope, make sure spring bean doesn't have shared instance variables otherwise it might lead to data inconsistency issues because it's not thread-safe.
- **Prototype**: A new instance will be created every time the bean is requested.
- **Request**: This is same as prototype scope, however it's meant to be used for web applications. A new instance of the bean will be created for each HTTP request.
- **Session**: A new bean will be created for each HTTP session by the container.
- **Global-session**: This is used to create global session beans for Portlet applications.

In case you are facing any challenges with these java interview questions, please comment on your problems in the section below.

Q5. Explain the role of DispatcherServlet and ContextLoaderListener.

DispatcherServlet is basically the front controller in the Spring MVC application as it loads the spring bean configuration file and initializes all the beans that have been

configured. If annotations are enabled, it also scans the packages to configure any bean annotated with @Component, @Controller, @Repository or @Service annotations.

ContextLoaderListener, on the other hand, is the listener to start up and shut down the WebApplicationContext in Spring root. Some of its important functions includes tying up the lifecycle of Application Context to the lifecycle of the ServletContext and automating the creation of ApplicationContext.

Q6. What are the differences between constructor injection and setter injection?

No.	Constructor Injection	Setter Injection
1)	No Partial Injection	Partial Injection
2)	Doesn't override the setter property	Overrides the constructor property if both are
3)	Creates a new instance if any modification occurs	Doesn't create a new instance if you change property value
4)	Better for too many properties	Better for a few properties.

Q7. What is autowiring in Spring? What are the autowiring modes?

Autowiring enables the programmer to inject the bean automatically. We don't need to write explicit injection logic. Let's see the code to inject bean using dependency injection.

1. <bean id="emp" class="com.javatpoint.Employee" autowire="byName" />

The autowiring modes are given below:

No.	Mode	Description
1)	no	this is the default mode, it means autowiring is not enabled.
2)	byName	Injects the bean based on the property name. It uses setter method.
3)	byType	Injects the bean based on the property type. It uses setter method.
4)	constructor	It injects the bean using constructor

Q8. How to handle exceptions in Spring MVC Framework?

Spring MVC Framework provides the following ways to help us achieving robust exception handling.

Controller Based:

We can define exception handler methods in our controller classes. All we need is to annotate these methods with @ExceptionHandler annotation.

Global Exception Handler:

Exception Handling is a cross-cutting concern and Spring provides @ControllerAdvice annotation that we can use with any class to define our global exception handler.

HandlerExceptionResolver implementation:

For generic exceptions, most of the times we serve static pages. Spring Framework provides HandlerExceptionResolver interface that we can implement to create global exception handler. The reason behind this additional way to define global exception handler is that Spring framework also provides default implementation classes that we can define in our spring bean configuration file to get spring framework exception handling benefits.

Q9. What are some of the important Spring annotations which you have used?

Some of the Spring annotations that I have used in my project are:

@**Controller** – for controller classes in Spring MVC project.

@**RequestMapping** – for configuring URI mapping in controller handler methods. This is a very important annotation, so you should go through Spring MVC RequestMapping Annotation Examples

@**ResponseBody** – for sending Object as response, usually for sending XML or JSON data as response.

@**PathVariable** – for mapping dynamic values from the URI to handler method arguments.

@**Autowired** – for autowiring dependencies in spring beans.

@**Qualifier** – with @Autowired annotation to avoid confusion when multiple instances of bean type is present.

@**Service** – for service classes.

@**Scope** – for configuring the scope of the spring bean.

@**Configuration, @ComponentScan and @Bean** – for java based configurations.

AspectJ annotations for configuring aspects and advices , @Aspect, @Before, @After, @Around, @Pointcut, etc.

Q10. How to integrate Spring and Hibernate Frameworks?

We can use Spring ORM module to integrate Spring and Hibernate frameworks if you are using Hibernate 3+ where SessionFactory provides current session, then you should avoid using HibernateTemplate or HibernateDaoSupport classes and better to use DAO pattern with dependency injection for the integration.

Also, Spring ORM provides support for using Spring declarative transaction management, so you should utilize that rather than going for hibernate boiler-plate code for transaction management.

Q11. Name the types of transaction management that Spring supports.

Two types of transaction management are supported by Spring. They are:

1. **Programmatic transaction management:** In this, the transaction is managed with the help of programming. It provides you extreme flexibility, but it is very difficult to maintain.
2. **Declarative transaction management:** In this, transaction management is separated from the business code. Only annotations or XML based configurations are used to manage the transactions.

Apart from these Core Java interview questions for experienced professionals, if you want to get trained by professionals on this technology, you can opt for a structured training from edureka!