By OnlineInterviewQuestions.com

Spring Interview Questions

Spring is an open source development framework for enterprise Java. The main features of spring may be used for developing applications of Java. Also, extensions for making web applications on top of JavaEE platform can be done. The main goal of spring is to make the development of J2EE development a lot easier to use and promote a programming practice by making use of POJO-based models. Spring is a lightweight framework. It can also be termed as the framework of frameworks because it also provides to other frameworks like EJB, JSF, Tapestry, Hibernate etc. This framework is defined as a structure using which we can find the solutions to various technical problems. It comprises many modules like ORM, DAO, AOP, WEB MVC etc.

Read Best Spring Interview Questions for Experienced Professionals

We hereby provide you with some questions on spring. These **Spring Interview Questions** have been specially designed so that it will help you to understand the nature of questions you may encounter while you are in an interview and the subject is spring.

In many interviews, interviewers start questioning the basic concept and then ask questions based on how you answer the basic ones.

Q1. What are the features/ characteristics of spring?

The important features of spring are as follows:

- **Lightweight**: Considering transparency and size, spring is lightweight. The basic versions of spring are below 2MB in size. Also, the overhead processing is also quite negligible.
- **Inversion of control (IOC)**: The objects in spring provide their dependencies instead of creating dependent objects. This is known as Inversion of Control.
- **Aspect-oriented Programming (AOP):** Aspect-oriented programming or AOP in spring supports cohesive development. Separating application and business logic from system services does this.
- **Container**: Spring creates as well as manages the life cycle and configures the application objects.
- **MVC Framework:** Spring's MVC application framework can be configured easily with the help of strategy interfaces, and it also accommodates multiple view technologies like JSP, POI, and iText. Therefore different frameworks can be used instead of Spring Framework (MVC).
- **Transaction Management:** Spring comes with Generic abstraction layer for transaction management. This allows us to add the many transaction managers, which make it easy to making transactions without thinking much about low-level issues.
- **JDBC Exception Handling**: The abstraction layer JDBC of Spring provides an exception hierarchy, which makes the error handling strategy simple.

Q2. How many types of modules are there in spring? Name them.

There are around twenty modules, which can be categorized into Core Container, Web, Data Access or Integration, Aspect Oriented Programming and Instrumentation and Test.

The modules are:

- Core module
- Bean module
- SpEL module
- Context module
- JDBC (Java DataBase Connectivity)
- ORM (Object Relational Mapping)
- OXM (Object XML Mappers)
- JMS (Java Messaging Service)
- Transaction
- Web
- Web MVC
- Web Socket
- Web Portlet
- Aspect Oriented Programming (AOP)
- Instrumentation
- Test
- Messaging
- Aspects

Q3. What do you mean by Spring Java Based Configuration?

Spring configuration files are XML files. These files contain the information of classes and describe how these classes can be configured and merged to each other. Java based configuration is a feature which enables the user to write most of their spring configuration without using XML. Instead, they are written with the help Java-based annotations.
For example, Java Annotation @Configuration is used to indicate that Spring IoC container can use the class, which is a source of bean definitions.

Q4. What are the advantages of spring?

There are many advantages of spring. Some of them are as follows:

1. Spring provides for templates for JDBC, Hibernate, and other technologies. Thus there's no need of writing many codes. It encapsulates the basic steps in the given technologies.
2. Spring applications are coupled loosely because of interdependency.
3. The Dependency Injection provided by spring makes it easier to test any application.
4. Spring is lightweight in terms of size and transparency.
5. The Dependency Injection feature provided by spring makes the development of Java EE application easy.
6. Spring provides power abstraction to Java-like JDBC, JPA, JTA, and JMS.
7. Spring also provides caching, formatting, validating and transactions.

Q5. What are the various components of a spring application?

A spring application consists of the following components:

- **Interface**: The functions in spring are defined by the interface.
- **Bean class:** It consists of the properties, its setter-getter methods, other functions etc.
- **Spring Aspect Oriented Programming (AOP):** In spring AOP provides for the functionality of crosscutting concerns.
- **User program:** It uses the function to implement.

Q6. What are the various ways in which Spring Framework can be used?

There are many ways in which Spring Framework can be used. They are listed as follows:

- It can be used as a full time Spring web application.
- Using Spring Frameworks middle-tier, it can be used as a third-party web framework.
- It can be used for remote usage.
- It can be used as Enterprise Java Bean which has the capability to wrap existing Plain Old Java Objects (POJO)

Q7. What do you mean by Spring IOC Container?

At the center of the Spring Framework, lies the spring container. It is the container, which creates the object, wires them together, makes necessary changes in them and manages their entire life cycle. The spring container uses the Dependency Injection to manage the elements that the application is made up of. The container gets the instructions for which objects to assemble, instantiate and configure and by reading the modification metadata provided. This metadata is provided either by XML, Java code or Java annotations.

Q8. What are the types of Dependency Injection supported by spring?

There is two Dependency Injection that is supported by spring.

- **Setter Injection :** Setter based Dependency Injection can be initialized by calling setter methods on the beans of the user after starting a no argument constructor to initialize their bean.
- **Constructor Injection:** Constructor based Dependency Injection is initialized by starting a constructor with a number of arguments, each working as a collaborator.

Q9. What are the differences between constructor injection and setter injection?

The difference between Constructor Injection and Setter Injection are:

- In constructor injection, there’s no partial injection while setter injection provides for partial injection.
- Constructor injection overrides the setter property while setter injection overrides the constructor property.
- Constructor injection creates a new instance for any kind of modification while setter injection doesn’t.
- Constructor injection works better for many properties while setter injection works better for few

properties.

Q10. What are the differences between Bean Factory and Application Context?

The differences between Bean Factory and Application Context are as follows:

- Bean Factory is an interface defined in org.springframework.beansfactory.BeanFactory, while Application Context is an interface defined, is org.springframework.context.ApplicationContext
- While the former uses lazy initialization the latter uses aggressive initialization
- While the former explicitly provide for resource object using the syntax the latter creates and manages resource objects on its own
- The former doesn't support internationalization while the latter does.
- The former doesn't support annotation-based dependency while the latter does.

Q11. What do you mean by AOP?

AOP or Aspect oriented programming is a technique in programming with the help of which programmers can modularize the crosscutting concerns or change the behavior that cuts across the various divisions of responsibility. Some of the examples of crosscutting concerns are logging well as transaction management. The center of AOP is an aspect. This encapsulates behaviors that can affect multiple classes into modules that can be reused.

Q12. What is an Advice? What are the different types of Advices?

An Action, which is taken by any aspect at a particular join point, is called an Advice. AOPs use an advice as an interceptor that maintains a train of interceptors near the join point.
Various types of advice are as follows:

- **Before**: These are types of advices, which get executed before the join point methods and can be configured using @Before annotation mark.
- **After returning:** These are the types of advices which get executed after the join point methods completes executing and the annotation mark used to configure it is @AfterReturning
- **After throwing:** These are the types of advices that execute only and only if join point method returns by exiting an exception and annotation mark used to configure it is @AfterThrowing.
- **After (finally):** These are the types of advices which gets executed after a join point method, not concerning whether the method's exit normally or exceptional return and it can be configured using @After annotation mark.
- **Around:** These are the types of advices that get executed before and after a join point and can be configured using the @Around annotation mark.

Q13. What do you mean by spring beans? What does bean scopes spring support? Explain each.

The objects, which are most important for the user's application and are managed by the containers of Spring IoC are called beans. A bean is an object that is initialized, put together, and is managed by a Spring IoC

container. The beans are created with the help of configuration metadata that are supplied to the container by the users.

The Spring Framework makes use of five scopes. Three of these scopes are valid only if it is in the form of a web-aware Application Context.

- **Singleton**: This is the scope that the bean defines to a single instance for each Spring IoC container.
- **Prototype**: This is the scopes that a bean defines to have any number of object instances.
- **Request:** This is the scopes that a bean defines as per HTTP requests. This is valid in the form of a web-aware Spring Application Context
- **Session**: This is the scopes that a bean defines to an HTTP session. This is valid in the form of a web-aware Spring Application Context
- **Global-session**: This is the scopes that a bean defines to a global HTTP session. This is valid in the form of a web-aware Spring Application Context.

Q14. In how many ways, can one configure Spring into an application?

One can configure spring into an application using three different ways. They are:

- Using XML Configuration
- Using Annotation configuration
- Using Java configuration

Q15. What are the disadvantages of auto wiring?

Following are the disadvantages of auto wiring:

- **Overriding possibility** – Despite using auto wiring one can still use dependencies like <property> and <constructor-arg> settings, which always can override auto wiring.
- **Primitive data types** ? one cannot auto wire some simple properties like classes, strings, and primitives etc.
- **Confusing nature** – Auto wiring is less exact than explicit wiring and thus it is not preferred if explicit wiring can be used.

Q16. What is an Aspect?

A module in spring which contains a set of APIs, which provides for crosscutting requirements is an Aspect. For example, a module that is used in logging is called AOP aspect for logging. Any application can have as many aspects as per requirement.

Q17. What is a Join point?

Join point is a point in an application where one can initialize an AOP aspect. One can also define it as the exact place in the application where an action is said to take place while using spring framework.

Q18. What is a bean definition made up of?

The bean definition is made of the information called configuration metadata, which is required by the container to have knowledge about the followings ?

- How bean can be created
- Details about the bean's lifecycle

On what factors Bean's lifecycle depends

Q19. What do you mean by Pointcut?

Pointcut is a collection of one or more join points where one advice can be executed. One can specify pointcut using patterns as well as expressions.

Q20. What does Weaving mean?

Weaving is the process by which aspects can be linked with other application objects for creating the required object.

Q21. How to use spring boot to configure different datasources ?

Q22. What is difference between Getmapping and Postmapping in Spring?

Major difference between Getmapping and Postmapping

Getmapping: Getmapping is a Spring notation and is widely used in mapping HTTP GET requests onto some specific handler methods. Getmapping is not generally used in mapping handler classes. This feature differentiates getmapping and requestmapping annotation from each other. It can be considered as an annotation that acts as a shortcut of requestmapping annotation. Getmapping requests handler paths onto specific handler methods. The lines of code are comparatively less than that of requestmapping.

Its request method can be framed as method= RequestMethod.GET. Getmapping is a newer annotation that has been developed to overcome the drawbacks of the previous requestmapping. It supports several attributes such as consume like @requestmapping. It is used to determine or map the GET & POST requests together and its method attribute is also not specified as getmapping maps to HTTP Get method always. This is composed of notation that acts as a shortcut for @requestmapping (method = RequestMethod.Get) it is used to get the requests on well defined and specific handler methods. In this notation it supports consumes where the options are-

```
consumes = "text/plain"
consumes = {"text/plain", "application/*"}
```

Postmapping: The latest version of Spring MVC known as Spring 4.3 bought with it some annotations, and postmapping is one of them. It has the same purpose as @requestmapping . Postmapping along with other new annotations is meta annotated. These annotations are meta-annotated with @requestmapping as per the related value of the "method " element. Postmapping depicts a combination of notation which is also an abbreviation for @requestmapping.

It is used to map out the Get requests only and its method attribute is also not specified as postmapping always maps HTTP post methods. It is composed of a notation that acts as a shortcut for @requestmapping (method= RequestMethod. Post). When we talk about the shortcut for @requestmapping we talk about @postmapping, it is a composed annotation that can act as it's shortcut as @ReqeustMapping(method= RequestMethod.POST).

Variants of **@requestmapping** other than **@getmapping** and **@postmapping** are **@patchmapping**, **@putmapping,** and **@deletemapping**.

Q23. What is an aspect in spring?

In Spring, an **aspect** is a class that performs enterprise application interests that cut across various classes, such as transaction management and its role looks and smells like it should have structure, but you can't find a way to represent this structure in code with conventional object-oriented techniques. Aspects can be a regular class configured through Spring XML configuration or we can use Spring AspectJ integration to define a class as Aspect employing **@Aspect annotation**.