**Spring And Spring Boot Interview Questions**

**Spring framework :-**

Spring is light weight ,open source framework meant for reducing the complexity of developing enterprise-level- application. It is also called “Framework of Frameworks”, as spring provides support to various other important framework.

Spring is a Dependency Injection framework to make java Application ‘Loosely Coupled”.

-Loose coupling allows us making changes in the code easily.

- Testing of loosely coupled structures is easier then tightly coupled structures.

What is Spring Boot ? Why did you use Spring boot in your project ? Why not Spring ?

Spring Boot is a Spring Module. Spring boot is a framework for Rapid Application Development build using Spring framework with extra support of Auto-Configuration and embedded application Server like Tomcat, Jetty .

Why :- Because it provides “Rapid Application Development”. Spring requires much of the boiler plate codes for configurations and dependency injections, all those things removed in spring boot. Hence , we can use spring boot rather than just spring framework.

What is the use of profiles in Spring Boot ?

- Suppose we create a Spring boot Application , that time we deal with different environments (like Development, Testing, production).

In development environment – we use H2 database & in production environment we use Oracle/ MySQL. In this case Setting will be change.

So for that , Spring has the provision of profiles to keep the separate configuration of each Environments.

**What are the features of Spring Framework?**

* Spring framework follows **layered architecture** pattern that helps in the necessary components selection along with providing a robust and cohesive framework for J2EE applications development.
* The AOP (Aspect Oriented Programming) part of Spring supports unified development by ensuring **separation of application’s business logic** from other system services.
* Spring provides **highly configurable** MVC web application framework which has the ability to switch to other frameworks easily.
* Provides provision of **creation and management** of the configurations and defining the lifecycle of application objects.
* Spring has a special design principle which is known as IoC (**Inversion of Control**) that supports objects to give their dependencies rather than looking for creating dependent objects.
* Spring is a **lightweight, java based, loosely coupled** framework.
* Spring provides generic **abstraction layer for transaction management** that is also very useful for container-less environments.
* Spring provides a convenient API to translate technology-specific exceptions (thrown by JDBC, Hibernate or other frameworks) into **consistent, unchecked exceptions.** This introduces abstraction and greatly simplifies exception handling.

**What is Spring Framework?**

* Spring is a powerful open-source, loosely coupled, lightweight, [java framework](https://www.interviewbit.com/java-interview-questions/) meant for reducing the complexity of developing enterprise-level applications. This framework is also called the “framework of frameworks” as spring provides support to various other important frameworks like JSF, Hibernate, Structs, EJB, etc.
* There are around 20 modules which are generalized into the following types:
  + Core Container
  + Data Access/Integration
  + Web
  + AOP (Aspect Oriented Programming)
  + Instrumentation
  + Messaging
  + Test

### What is a Spring configuration file?

A Spring configuration file is basically an XML file that mainly contains the classes information and describes how those classes are configured and linked to each other. The XML configuration files are verbose and cleaner.

### What do you mean by IoC (Inversion of Control) Container?

Spring container forms the core of the Spring Framework. The Spring container uses Dependency Injection (DI) for managing the application components by creating objects, wiring them together along with configuring and managing their overall life cycles. The instructions for the spring container to do the tasks can be provided either by XML configuration, Java annotations, or Java code.

**What do you understand by Dependency Injection?**

-The main idea in Dependency Injection is that you don’t have to create your objects but you just have to describe how they should be created.

* The components and services need not be connected by us in the code directly. We have to describe which services are needed by which components in the configuration file. The IoC container present in Spring will wire them up together.

In Java, the 2 major ways of achieving dependency injection are:

* Constructor injection: Here, the IoC container invokes the class constructor with a number of arguments where each argument represents a dependency on the other class.
* Setter injection: Here, the spring container calls the setter methods on the beans after invoking a no-argument static factory method or default constructor to instantiate the bean.

**Explain the difference between constructor and setter injection?**

* In constructor injection, partial injection is not allowed whereas it is allowed in setter injection.
* The constructor injection doesn’t override the setter property whereas the same is not true for setter injection.
* Constructor injection creates a new instance if any modification is done. The creation of a new instance is not possible in setter injection.
* In case the bean has many properties, then constructor injection is preferred. If it has few properties, then setter injection is preferred.
* **What are Spring Beans?**
* They are the objects forming the backbone of the user’s application and are managed by the Spring IoC container.
* Spring beans are instantiated, configured, wired, and managed by IoC container.
* Beans are created with the configuration metadata that the users supply to the container (by means of XML or java annotations configurations.)
* **What are the bean scopes available in Spring?**

The Spring Framework has five scope supports. They are:

* **Singleton:** The scope of bean definition while using this would be a single instance per IoC container.
* **Prototype:** Here, the scope for a single bean definition can be any number of object instances.
* **Request:**The scope of the bean definition is an HTTP request.
* **Session:** Here, the scope of the bean definition is HTTP-session.
* **Global-session:** The scope of the bean definition here is a Global HTTP session.

**What is autowiring and name the different modes of it?**

The IoC container autowires relationships between the application beans. Spring lets collaborators resolve which bean has to be wired automatically by inspecting the contents of the BeanFactory.  
Different modes of this process are:

* **no**: This means **no autowiring** and is the default setting. An explicit bean reference should be used for wiring.
* **byName**: The bean dependency is injected according to the **name of the bean**. This matches and wires its properties with the beans defined by the same names as per the configuration.
* **byType**: This injects the bean dependency based on **type**.
* **constructor**: Here, it injects the bean dependency **by calling the constructor** of the class. It has a large number of parameters.
* **autodetect**: First the container tries to wire using autowire by the constructor, if it isn't possible then it tries to autowire by byType.

**What are the limitations of autowiring?**

* **Overriding possibility**: Dependencies are specified using <constructor-arg> and <property>  settings that override autowiring.
* **Data types restriction**: Primitive data types, Strings, and Classes can’t be autowired

Spring Boot Interview Questions

### What do you understand by the term ‘Spring Boot’?

Spring Boot is an open-source, java-based framework that provides support for Rapid Application Development and gives a platform for developing stand-alone and production-ready spring applications with a need for very few configurations

**Explain the advantages of using Spring Boot for application development.?**

* Spring Boot helps to create stand-alone applications which can be started using java.jar (Doesn’t require configuring WAR files).
* Spring Boot also offers pinpointed ‘started’ POMs to Maven configuration.
* Has provision to embed Undertow, Tomcat, Jetty, or other web servers directly.
* Auto-Configuration: Provides a way to automatically configure an application based on the dependencies present on the classpath.
* Spring Boot was developed with the intention of lessening the lines of code.
* It offers production-ready support like monitoring and apps developed using spring boot are easier to launch

**Differentiate between Spring and Spring Boot.**

* The Spring Framework provides multiple features like dependency injection, data binding, aspect-oriented programming (AOP), data access, and many more that help easier development of web applications whereas Spring Boot helps in easier usage of the Spring Framework by simplifying or managing various loosely coupled blocks of Spring which are tedious and have a potential of becoming messy.
* Spring boot simplifies commonly used spring dependencies and runs applications straight from a command line. It also doesn’t require an application container and it helps in monitoring several components and configures them externally.

### What are the Spring Boot key components?

Below are the four key components of spring-boot:

* Spring Boot auto-configuration.
* Spring Boot CLI.
* Spring Boot starter POMs.
* Spring Boot Actuators

**What are the features of Spring Boot?**

* **Spring Boot CLI** – This allows you to Groovy / Maven for writing Spring boot application and avoids boiler-plate code.
* **Starter Dependency** – With the help of this feature, Spring Boot aggregates common dependencies together and eventually improves productivity and reduces the burden on
* **Spring Initializer**– This is a web application that helps a developer in creating an internal project structure. The developer does not have to manually set up the structure of the project while making use of this feature.
* **Auto-Configuration** – This helps in loading the default configurations according to the project you are working on. In this way, unnecessary WAR files can be avoided.
* **Spring Actuator** – Spring boot uses actuator to provide “Management EndPoints” which helps the developer in going through the Application Internals, Metrics etc.
* **Logging and Security** – This ensures that all the applications made using Spring Boot are properly secured without any hassle.
* **What are the Spring Boot key components?**

Below are the four key components of spring-boot:

* Spring Boot auto-configuration.
* Spring Boot CLI.
* Spring Boot starter POMs.
* Spring Boot Actuators.

**Why Spring Boot over Spring?**

Below are some key points which spring boot offers but spring doesn’t:

* Starter POM.
* Version Management.
* Auto Configuration.
* Component Scanning.
* Embedded server.
* InMemory DB.
* Actuators

### What does the @SpringBootApplication annotation do internally?

The @SpringBootApplication annotation is equivalent to using @Configuration, @EnableAutoConfiguration, and @ComponentScan with their default attributes. Spring Boot enables the developer to use a single annotation instead of using multiple. But, as we know, Spring provided loosely coupled features that we can use for each annotation as per our project needs.

### What is Spring Boot CLI and what are its benefits?

Spring Boot CLI is a command-line interface that allows you to create a spring-based java application using Groovy.

Example: You don’t need to create getter and setter method or access modifier, return statement. If you use the JDBC template, it automatically loads for you.

### Explain @RestController annotation in Sprint boot?

It is a combination of @Controller and @ResponseBody, used for creating a restful controller. It converts the response to JSON or XML. It ensures that data returned by each method will be written straight into the response body instead of returning a template.

### What is the difference between @RestController and @Controller in Spring Boot?

@Controller Map of the model object to view or template and make it human readable but @RestController simply returns the object and object data is directly written in HTTP response as JSON or XML.

What is the difference between RequestMapping and GetMapping?

RequestMapping can be used with GET, POST, PUT, and many other request methods using the method attribute on the annotation. Whereas getMapping is only an extension of RequestMapping which helps you to improve on clarity on request.

### What is the use of Profiles in spring boot?

While developing the application we deal with multiple environments such as dev, QA, Prod, and each environment requires a different configuration. For eg., we might be using an embedded H2 database for dev but for prod, we might have proprietary Oracle or DB2. Even if DBMS is the same across the environment, the URLs will be different.

To make this easy and clean, Spring has the provision of Profiles to keep the separate configuration of environments.

### What is Spring Actuator? What are its advantages?

An actuator is an additional feature of Spring that helps you to monitor and manage your application when you push it to production. These actuators include auditing, health, CPU usage, HTTP hits, and metric gathering, and many more that are automatically applied to your application.

**What is Spring AOP?**

* Spring AOP (Aspect Oriented Programming) is similar to OOPs (Object Oriented Programming) as it also provides modularity.
* In AOP key unit is **aspects** or **concerns** which are nothing but stand-alone modules in the application. Some aspects have centralized code but other aspects may be scattered or tangled code like in the case of logging or transactions. These scattered aspects are called **cross-cutting concern**.
  + A cross-cutting concern such as transaction management, authentication, logging, security etc is a concern that could affect the whole application and should be centralized in one location in code as much as possible for security and modularity purposes.
* AOP provides platform to dynamically add these cross-cutting concerns before, after or around the actual logic by using simple pluggable configurations.
* This results in easy maintainenance of code. Concerns can be added or removed simply by modifying configuration files and therefore without the need for recompiling complete sourcecode.
* There are 2 types of implementing Spring AOP:
  + Using XML configuration files
  + Using AspectJ annotation style.

**. What are the benefits of Spring MVC framework over other MVC frameworks?**

* Clear separation of roles – There is a specialised dedicated object for every role.
* Reusable business code logic – With Spring MVC, there is no need for duplicating the code. Existing objects can be used as commands instead of replicating them in order to extend a particular framework base class.
* Spring MVC framework provides customizable binding and validation.
* Also provides customizable locale and theme resolution.
* Spring MVC supports customizable handler mapping and view resolution too.

# **IoC Container**

1. [IoC Container](https://www.javatpoint.com/ioc-container)
2. [Using BeanFactory](https://www.javatpoint.com/ioc-container)
3. [Using ApplicationContext](https://www.javatpoint.com/ioc-container)

The IoC container is responsible to instantiate, configure and assemble the objects. The IoC container gets informations from the XML file and works accordingly. The main tasks performed by IoC container are:

* to instantiate the application class
* to configure the object
* to assemble the dependencies between the objects

There are two types of IoC containers. They are:

1. **BeanFactory**
2. **ApplicationContext**