What is Spring?

Spring is an open source development framework for enterprise Java. The core features of the Spring Framework can be used in developing any Java application, but there are extensions for building web applications on top of the Java EE platform. Spring framework targets to make J2EE development easier to use and promote good programming practice by enabling a POJO-based programming model.

What are benefits of using spring?

Following is the list of few of the great benefits of using Spring Framework:

* **Lightweight** − Spring is lightweight when it comes to size and transparency. The basic version of spring framework is around 2MB.
* **Inversion of control (IOC)** − Loose coupling is achieved in spring using the technique Inversion of Control. The objects give their dependencies instead of creating or looking for dependent objects.
* **Aspect oriented (AOP)** − Spring supports Aspect oriented programming and enables cohesive development by separating application business logic from system services.
* **Container** − Spring contains and manages the life cycle and configuration of application objects.
* **MVC Framework** − Spring's web framework is a well-designed web MVC framework, which provides a great alternative to web frameworks such as Struts or other over engineered or less popular web frameworks.
* **Transaction Management** − Spring provides a consistent transaction management interface that can scale down to a local transaction (using a single database, for example) and scale up to global transactions (using JTA, for example).

What is Dependency Injection?

We don’t need to create your objects but describe how they should be created. You don't directly connect your components and services together in code but describe which services are needed by which components in a configuration file. A container (the IOC container) is then responsible for hooking it all up.

What are the different types of IoC (dependency injection)?

Types of IoC are −

* **Constructor-based dependency injection** − Constructor-based DI is accomplished when the container invokes a class constructor with a number of arguments, each representing a dependency on other class.
* **Setter-based dependency injection** − Setter-based DI is accomplished by the container calling setter methods on your beans after invoking a no-argument constructor or no-argument static factory method to instantiate your bean.

What are Spring beans?

. A bean is an object that is instantiated, assembled, and otherwise managed by a Spring IoC container.

What bean scopes does Spring support? Explain them.

The Spring Framework supports following five scopes, three of which are available only if you use a web-aware ApplicationContext.

* **singleton** − This scopes the bean definition to a single instance per Spring IoC container.
* **prototype** − This scopes a single bean definition to have any number of object instances.
* **request** − This scopes a bean definition to an HTTP request. Only valid in the context of a web-aware Spring ApplicationContext.
* **session** − This scopes a bean definition to an HTTP session. Only valid in the context of a web-aware Spring ApplicationContext.
* **global-session** − This scopes a bean definition to a global HTTP session. Only valid in the context of a web-aware Spring ApplicationContext.

What is default scope of bean in Spring framework?

The default scope of bean is **Singleton** for Spring framework.

What does @Autowired annotation mean?

Injecting a bean into an another spring bean, which reduces the tight coupling where spring automatically resolve the instance

What does @Qualifier annotation mean?

There may be a situation when you create more than one bean of the same type and want to wire only one of them with a property, in such case you can use @Qualifier annotation along with @Autowired to remove the confusion by specifying which exact bean will be wired.

What is a DispatcherServlet?

The Spring Web MVC framework is designed around a DispatcherServlet that handles all the HTTP requests and responses.

What is Controller in Spring MVC framework?

Controllers provide access to the application behavior that you typically define through a service interface. Controllers interpret user input and transform it into a model that is represented to the user by the view. Spring implements a controller in a very abstract way, which enables you to create a wide variety of controllers.

Explain the *@Controller* annotation.

The *@Controller* annotation indicates that a particular class serves the role of a controller. Spring does not require you to extend any controller base class or reference the Servlet API.

Explain *@RequestMapping* annotation.

*@RequestMapping* annotation is used to map a URL to either an entire class or a particular handler method.

[**Spring**](https://www.edureka.co/blog/spring-tutorial/)**vs Spring Boot**

|  |  |
| --- | --- |
| **Spring** | **Spring** Boot |
| A web application framework based on Java | A module of Spring |
| Provides tools and libraries to create customized web applications | Used to create a Spring application project which can just run/ execute |
| Spring is more complex than Spring Boot | Spring Boot is less complex than the Spring framework |
| Need to deploy in a containerr | Default container is provided |

## ****Mention the advantages of Spring Boot****

The advantages of Spring Boot are as follows:

Provides auto-configuration to load a set of default configuration for a quick start of the application

Creates stand-alone applications with a range of non-functional features that are common to large classes of projects

It comes with embedded tomcat, servlet containers jetty to avoid the usage of WAR files

* Spring Boot provides an opinionated view to reduce the developer effort and simplify maven configurations
* Provides CLI tool to develop and test applications
* Comes with Spring Boot starters to ensure dependency management and also provides various security metrics
* Consists of a wide range of APIs for monitoring and managing applications in dev and prod.
* Integrates with Spring Ecosystem like Spring [JDBC](https://www.edureka.co/blog/connect-mysql-database-in-java), Spring ORM, Spring Data, Spring Security easily by avoiding boilerplate code

**Application.Properties or application.yml**

## ****Mention the differences between JPA and****[Hibernate](https://www.edureka.co/blog/what-is-hibernate-in-java/)

|  |  |
| --- | --- |
| **JPA** | **Hibernate** |
| JPA is a Data Access Abstraction used to reduce the amount of boilerplate code | Hibernate is an implementation of Java Persistence API and offers benefits of loose coupling |