**1. 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 [Java EE development](https://intellipaat.com/tutorial/java-tutorial/introduction-java/) easier to use and promote good programming practice by enabling a POJO-based programming model.

**2. What are features of Spring?**

**Lightweight:**

Spring is lightweight when it comes to size and transparency. The basic version of spring framework is around 1MB. And the processing overhead is also very negligible.

**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 comes with [MVC web application framework](https://intellipaat.com/interview-question/java-interview-questions/), built on core Spring functionality. This framework is highly configurable via strategy interfaces, and accommodates multiple view technologies like JSP, Velocity, Tiles, iText, and POI. But other frameworks can be easily used instead of Spring MVC Framework.

**JDBC Exception Handling:**

The JDBC abstraction layer of the Spring offers a meaningful exception hierarchy, which simplifies the error handling strategy. Integration with Hibernate, JDO, and iBATIS: Spring provides best Integration services with Hibernate, JDO and iBATIS.

**Transaction Management:**

Spring framework provides a generic abstraction layer for transaction management. This allowing the developer to add the pluggable transaction managers, and making it easy to demarcate transactions without dealing with low-level issues. Spring’s transaction support is not tied to J2EE environments and it can be also used in container less environments.

**3. Describe the Spring Framework?**

The Spring Framework provides a comprehensive programming and configuration model for modern Java-based enterprise applications – on any kind of deployment platform. A [key element of Spring](https://intellipaat.com/interview-question/java-interview-questions/) is infrastructural support at the application level: Spring focuses on the “plumbing” of enterprise applications so that teams can focus on application-level business logic, without unnecessary ties to specific deployment environments.

**Get Spring Certification in just 15 Hours**

**GET CERTIFIED**

**4. What is Spring Java Based Configuration?**

Java based configuration option enables the user to write most of their Spring configuration without XML but with the help of few Java-based annotations.

for example:

Annotation **@Configuration** indicates that the class can be used by the Spring IoC container as a source of bean definitions. The **@Bean** annotation tells spring that a method annotated with @Bean will return an object that should be registered as a bean in the spring application context.

**5. Describe some of the standard Spring events?**

Spring provides the following standard events:

* **ContextRefreshedEvent**: This event is published when the ApplicationContext is either initialized or refreshed. This can also be raised using the refresh() method on the ConfigurableApplicationContext interface.
* **ContextStartedEvent:** This event is published when the ApplicationContext is started using the start() method on the ConfigurableApplicationContext interface. the user can poll their database or they can re/start any stopped application after receiving this event.
* **ContextStoppedEvent:** This event is published when the ApplicationContext is stopped using the stop() method on the ConfigurableApplicationContext interface. the users can do required housekeep work after receiving this event.
* **ContextClosedEvent:** This event is published when the ApplicationContext is closed using the close() method on the ConfigurableApplicationContext interface. A closed context reaches its end of life; it cannot be refreshed or restarted.
* **RequestHandledEvent:** This is a web-specific event telling all beans that an HTTP request has been serviced.

**Download Spring Interview questions asked by top MNCs in 2017 ?**

Top of Form



Bottom of Form

**6. Which are the Spring framework modules?**

The basic modules of the Spring framework are :

* Core module
* Bean module
* Context module
* Expression Language module
* JDBC module
* [ORM module](https://intellipaat.com/tutorial/datastage-tutorial/ibm-information-server-architecture/)
* OXM module
* Java Messaging Service(JMS) module
* Transaction module
* Web module
* Web-Servlet module
* Web-Struts module
* Web-Portlet module

Advanced Questions

**1. What are the types of Dependency Injection Spring supports?**

**Setter Injection**

Setter-based DI is realized by calling setter methods on the user’s beans after invoking a no-argument constructor or no-argument static factory method to instantiate their bean.

**Constructor Injection**

Constructor-based DI is realized by invoking a constructor with a number of arguments, each representing a collaborator.

**2. What is Spring IOC container?**

The Spring IOC creates the objects, wire them together, configure them, and manage their complete lifecycle from creation till destruction. The spring container uses dependency injection (DI) to manage the components that make up an application.

**3. What are Spring beans?**

The objects that form the backbone of the users application and that are managed by the Spring IoC container are called beans. A bean is an object that is instantiated, assembled, and otherwise managed by a Spring IoC container. These beans are created with the configuration metadata that the users supply to the container.

**4. What bean scopes does Spring support? Explain them.**

The Spring Framework supports following five scopes, three of which are available only if the users use a web-aware Application Context.

**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](https://intellipaat.com/interview-question/mysql-interview-questions/). 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.

**5. Explain Bean lifecycle in Spring framework?**

Following is sequence of a bean lifecycle in Spring:

**Instantiate:** First the spring container finds the bean’s definition from the XML file and instantiates the bean.

**Populate properties:** Using the dependency injection, spring populates all of the properties as specified in the bean definition.

**Set Bean Name:** If the bean implements BeanNameAware interface, spring passes the bean’s id to setBeanName() method.

**Set Bean factory:** If Bean implements BeanFactoryAware interface, spring passes the beanfactory to setBeanFactory() method.

**Pre Initialization:** Also called post process of bean. If there are any bean BeanPostProcessors associated with the bean, Spring calls postProcesserBeforeInitialization() method.

**Initialize beans:** If the bean implements IntializingBean,its afterPropertySet() method is called. If the bean has init method declaration, the specified initialization method is called.

**Post Initialization:**– If there are any BeanPostProcessors associated with the bean, their postProcessAfterInitialization() methods will be called.

**Ready to use:** Now the bean is ready to use by the application

**Destroy:** If the bean implements DisposableBean , it will call the destroy() method