Zach Karr

IS296 – Spring Boot

02/04/2023

1. In Spring, a Bean is an instantiated object that is assembled and managed by an inversion of control (IoC) container. A IoC is a process which allows us to define an objects dependencies by providing configuration data.
2. Dependency injection is used to create loosely coupled classes, interfaces, etc. Dependency injection achieves loose coupling by providing an object the dependencies of another object. There are three classes involved – the Client class, the Service class, and the Injector class. The Client is dependent on the Service class, and the Service class in turn provides a service to the Client class. The Injector class is then responsible for injecting the Service class into the Client. There are also three types of dependency injection. These are called Constructor, Setter/Property, and Interface injection. The difference between these three is in where the Injector class supplies dependencies to the Client. With Constructor injection, the injector supplies them to the Client’s constructor. Likewise, with Setter/Property injection, this is done so through a setter method of the Client. With Interface injection, the injector uses an interface that the Client implements, and setters that accepts the dependencies.
3. Inversion of Control is a process which allows us to define an objects dependencies by providing configuration data. It does this by moving the control of objects to a container/framework instead. A framework such as Spring Boot will use abstractions to enable the framework to take control of a programs flow. This allows for more modularity, decoupling, and better testing by isolating your programs components.
4. The ApplicationContext interface is a subinterface of the BeanFactory. These two interfaces combined represent the IoC container. The BeanFactory is the root interface which accesses the spring container and the ApplicationContext provides all the functionalilty of the BeanFactory.
5. The ApplicationContext interface gets its instructions from configuration metadata, such as XML, Java annotations and Java code.
6. Dependency Injection has many advantages and some disadvantages. Some of the advantages include flexibility of configuring a Client class, since only the Client’s behavior is fixed. You can modify and alter a Service classes behaviors without causing any breaking changes to the Client due to the loosely coupled architecture that dependency injection provides. There are also advantages for testing purposes. Since the architecture is loosely coupled, it allows us to easily create mock objects that you’d find much harder to create otherwise (mocking data is much harder in a tightly coupled system due to the fact that classes and object instances of a class is dependent on the class hierarchy. Making changes to a class that down the hierarchy of classes for testing purposes would in many cases require changes to the parent class of the class you are trying to test). Dependency Injection also removes many unneeded dependencies. For example, if you were to use Inheritance instead of Dependency Injection, you would create and define a class (the parent), then create and define a child class that extends the parent class, and in doing so the child inherits all of the methods from the parent class (even if they aren’t needed).
7. “Spring is a Container” refers to the fact that the spring framework provides a container known as the “Spring IoC container”, also known as the “Core Spring Container”. The IoC/Core Container is responsible for managing the components and services in our Java applications. The IoC container is responsible for instantiating, configuring and assembling the objects and uses dependency injection to manage the dependencies.
8. Java Configuration provides a more concise, readable and flexible way to configure Spring Beans over XML for many reasons. One reason is type safety. With Java Configuration the Java compiler is able to catch errors because it is written in Java code. On the otherhand, XML uses Strings and this can lead to runtime errors (due to misspellings, etc). XML is not very enjoyable to read and understand, as compared to Java code (which Java developers are already much more comfortable reading). Java configuration can be reused across multiple projects, while XML is specific to a particular project.
9. The lifecycle of a Bean are as follows: 1. Bean instantiation. 2. Bean Configuration. 3. Bean Post-Processing. 4. Bean Wiring. 5. Bean Initialization. 6. Bean Usage. 7. Bean Destruction. The Spring container creates an instance of the bean using a method such as constructor injection, etc., then the Spring container sets the bean properties and dependencies. After the properties are set, the container checks for any BeanPostProcessor interfaces which perform additional processing before use. The Spring container then creates a relation between beans, and makes sure dependencies are met. After this, Spring calls the initialization method of the bean before the bean is used. Once initialized, the container makes the bean available for other beans or for your application to use. Finally, when Spring container is closed, it invokes the destroy method for the bean (if it exists) and performs cleanup and removes it from the Spring container.
10. The @Bean annotation indicates that a method should return a bean that is managed by the container.
11. @Autowired specifies that a property, setter, or constructor should be wired/injected with a corresponding bean. The container automatically finds and injects a bean that matches the type of annotated property, setter or constructor.
12. The @Configuration annotation specifies that a class contains configuration information for the container. When using @Configuration, a class is then considered as a config class and can be used to define beans and dependencies.
13. The default scope of a bean is singleton, meaning that only a single instance of a class is created by the container.
14. Boilerplate code is repetitive, standard required code that is the foundation of an application, such as code for connecting to databases or defining a class’ structure.
15. Steve Jobs said that the way you increase programmer productivity is not by increasing the amount of lines of code a programmer has to write, but rather by eliminating as much as possible (his words are 80%). While Spring Boot does help reduce the amount of code a programmer has to write, and is a huge improvement over vanilla Java, I don’t think it reduces lines of code written by 80% when compared to vanilla Java (I tried finding statistics for this but I couldn’t find any). There is also a learning curve that comes with learning any new framework which can hinder programmer productivity initially, but an experienced programmer can pick up a new framework rather quickly. Overall, Spring does increase programmer productivity, although I am not sure that Spring would be what Steve Jobs had in mind when he gave that speech (Steve Jobs passed away in 2011 and Spring Boot was released in 2014, so we have no way of knowing what he would have thought).
16. Cross-cutting concerns refers to aspects of development that are relevant to multiple parts of an application, but have do not easily fit into one particular component.
17. Separations of Concerns is a design principle in development that says that a software system should be structured in a way that separates the different concerns or responsibilities into distinct parts, allowing developers to focus on each concern individually and build/maintain the software in a more modular/maintainable way. It breaks down complex systems into smaller parts through design patterns such as Model-View-Controller and encapsulating cross-cutting concerns into aspects.
18. Cross-cutting concerns cause code to be difficult to understand, maintain or modify. This happens because cross-cutting concerns lead to code duplication, tight coupling, and often have complex logic to deal with the various parts of the application that require similar logic but has to be modified for a specific problem.
19. Component scanning refers to the process of automatically detecting components in the ApplicationContext. It allows developers to define and manage beans without having to explicitly register the components in configuration.
20. Spring creates objects from bean definitions using IoC, by which the container manages the creation and lifecycle of the bean.
21. The pom.xml file is used to manage the dependencies and setting configurations of an application. It is the central configuration file for a project.
22. The IDE will complain that the instance of Cake (c) is missing a required argument (flavor)
23. The name of the variable that is a dependency is flavor.
24. It would be Constructor injection, since it would be injected as a parameter to the constructor for Cake.
25. The factory pattern is a creational pattern that solves a problem of creating objects without specifying the exact class of the object that is to be created.