SPRING BOOT ASSIGNMENT - 3

1) What is Loose Coupling?

Ans: Loose coupling is an approach to interconnecting the components in a system or network so that those components, also called elements, depend on each other to the least extent practicable.

2) What is a Dependency?

Ans: A dependency is a link amongst a project's terminal elements. A Java class has a dependency on another class, if it uses an instance of this class. ... For example, a class which accesses a logger service has a dependency on this service class.

3) What is IOC (Inversion of Control)?

Ans: IoC is a framework used to create and inject dependencies automatically and manages its lifecycle.

4) What is Dependency Injection?

Ans: Dependency Injection is a way of making hard coded dependencies, loosely coupled. This could be done in a couple different methods: by using parameter in a constructor or through a setter method.

5) Can you give few examples of Dependency Injection?

Ans:

DI via parameter in a constructor:

```
public class ThisClass {

private ThisDependency thisDependency;

public ThisClass (ThisDependency thisDependency) {

this. thisDependency = thisDependency;

}
}
```

DI via setter method:

```
public class ThisClass {

private Object thisDependency;

public Object getThisDependency () {

return thisDependency;

}
```

6) What is Auto Wiring?

Ans: Auto-wiring feature of spring framework enables you to inject the object dependency implicitly. It internally uses setter or constructor injection. Auto-wiring can't be used to inject primitive and string values. It works with reference only.

7) What are the important roles of an IOC Container?

Ans: The IoC container is responsible to instantiate, configure and assemble the objects. The IoC container gets information 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

8) What are Bean Factory and Application Context?

Ans: They are the type of Inversion of Control (IoC) controllers.

SPRING BOOT ASSIGNMENT - 3

9) Can you compare Bean Factory with Application Context?

Ans:

BEAN FACTORY	APPLICATION CONTEXT
1. It is an IoC container	1. The ApplicationContext interface is built on top of the BeanFactory interface.
2. It only performs basic functions	2. It adds some extra functionality than BeanFactory such as simple integration with Spring's AOP, message resource handling (for I18N), event propagation, application layer specific context (e.g. WebApplicationContext) for web application.
3. Usage: Resource resource=new ClassPathResource("applicationContext.xml"); BeanFactory factory=new XmlBeanFactory(resource);	**3. Usage:** ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");

10) How do you create an application context with Spring?

Ans: The ClassPathXmlApplicationContext class is the implementation class of ApplicationContext interface. We need to instantiate the ClassPathXmlApplicationContext class to use the ApplicationContext as given below:

ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");

The constructor of ClassPathXmlApplicationContext class receives string, so we can pass the name of the xml file to create the instance of ApplicationContext.

11) How does Spring know where to search for Components or Beans?

Ans: This part of "telling Spring where to search" is called a **Component Scan**. You define the packages that have to be scanned. Once you define a Component Scan for a package, Spring would search the package and all its sub packages for components/beans.

SPRING BOOT ASSIGNMENT - 3

12) What is a Component Scan?

Ans: component scan is one method of asking Spring to detect Spring-managed components. Spring needs the information to locate and register all the Spring components with the application context when the application starts. Spring can auto scan, detect, and instantiate components from pre-defined project packages.

13) How do you define a component scan in XML and Java Configurations?

Ans:

```
  @Configuration
      @ComponentScan("com.company") // search the com.company package for @Component
classes
      @ImportXml("classpath:com/company/data-access-config.xml")
      public class Config {
      }
```

14) How is it done with Spring Boot?

Ans:

```
import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;


@SpringBootApplication  // same as @Configuration @EnableAutoConfiguration
@ComponentScan

public class Application {


    public static void main(String[] args) {

        SpringApplication.run(Application.class, args);

    }


}
```

15) What does @Component signify?

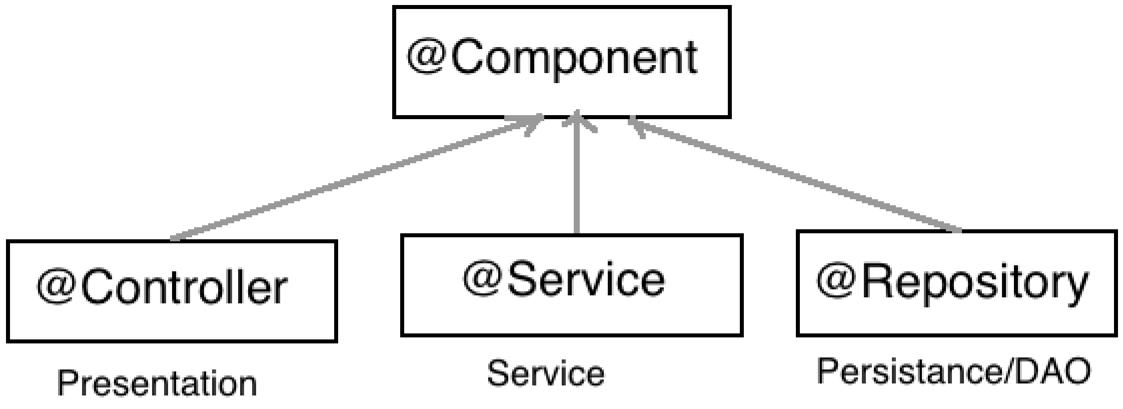
Ans: @Component annotation means that only a single instance of the annotated class gets created.

16) What does @Autowired signify?

Ans: @Autowired, Spring will look for a class that matches the property in the applicationContext, and inject it automatically. If you have more than one UserService bean, then you'll have to qualify which one it should use.

17) What's the difference Between @Controller, @Component, @Repository, and @Service Annotations in Spring?

Ans:



The difference between them is, @**component is** used to **annotate** compound classes, @**Repository is** a marker **for** automatic exception translation **in the** persistence layer, **for service** layer we need to use @**service**.

18) What is the default scope of a bean?

Ans: Singleton is the default scope for a Bean, the one that will be used if nothing else is indicated.

19) Are Spring beans thread safe?

Ans: Spring singleton beans are **NOT** thread-safe just because Spring instantiates them. Sorry. Spring just manage the life cycle of singleton bean and maintains single instance of object. Thread safety has nothing to do with it.

20) What are the other scopes available?

Ans: Types of Scopes:

- singleton(default*)
- prototype
- request
- session
- global session