18. 什么是Spring的依赖注入？

19. 有哪些不同类型的IOC（依赖注入）方式？

20. 哪种依赖注入方式你建议使用，构造器注入，还是 Setter方法注入？

21.什么是Spring beans?

22. 一个 Spring Bean定义包含什么？

23. 如何给Spring 容器提供配置元数据?

24. 你怎样定义类的作用域?

25. 解释Spring支持的几种bean的作用域。

26. Spring框架中的单例bean是线程安全的吗?

27. 解释Spring框架中bean的生命周期。

28. 哪些是重要的bean生命周期方法？你能重载它们吗？

29. 什么是Spring的内部bean？

30. 在 Spring中如何注入一个java集合？

31. 什么是bean装配?

32. 什么是bean的自动装配？

33. 解释不同方式的自动装配。

34.自动装配有哪些局限性 ?

35. 你可以在Spring中注入一个null 和一个空字符串吗？

|  |  |
| --- | --- |
|  |  |

**答案解析**ID:javaxxf

**18. 什么是Spring的依赖注入？**

依赖注入，是IOC的一个方面，是个通常的概念，它有多种解释。这概念是说你不用创建对象，而只需要描述它如何被创建。你不在代码里直接组装你的组件和服务，但是要在配置文件里描述哪些组件需要哪些服务，之后一个容器（IOC容器）负责把他们组装起来。

**19. 有哪些不同类型的IOC（依赖注入）方式？**

* 构造器依赖注入：构造器依赖注入通过容器触发一个类的构造器来实现的，该类有一系列参数，每个参数代表一个对其他类的依赖。
* Setter方法注入：Setter方法注入是容器通过调用无参构造器或无参static工厂方法实例化bean之后，调用该bean的setter方法，即实现了基于setter的依赖注入。

**20. 哪种依赖注入方式你建议使用，构造器注入，还是 Setter方法注入？**

你两种依赖方式都可以使用，构造器注入和Setter方法注入。最好的解决方案是用构造器参数实现强制依赖，setter方法实现可选依赖。

**21.什么是Spring beans?**

Spring beans 是那些形成Spring应用的主干的java对象。它们被Spring IOC容器初始化，装配，和管理。这些beans通过容器中配置的元数据创建。比如，以XML文件中<bean/> 的形式定义。

Spring 框架定义的beans都是单件beans。在bean tag中有个属性”singleton”，如果它被赋为TRUE，bean 就是单件，否则就是一个 prototype bean。默认是TRUE，所以所有在Spring框架中的beans 缺省都是单件。

**22. 一个 Spring Bean 定义包含什么？**

一个Spring Bean 的定义包含容器必知的所有配置元数据，包括如何创建一个bean，它的生命周期详情及它的依赖。

**23. 如何给Spring 容器提供配置元数据?**

这里有三种重要的方法给Spring 容器提供配置元数据。

XML配置文件。

基于注解的配置。

基于java的配置。

**24. 你怎样定义类的作用域?**

当定义一个<bean> 在Spring里，我们还能给这个bean声明一个作用域。它可以通过bean 定义中的scope属性来定义。如，当Spring要在需要的时候每次生产一个新的bean实例，bean的scope属性被指定为prototype。另一方面，一个bean每次使用的时候必须返回同一个实例，这个bean的scope 属性必须设为 singleton。

**25. 解释Spring支持的几种bean的作用域。**

Spring框架支持以下五种bean的作用域：

* **singleton**: bean在每个Spring ioc 容器中只有一个实例。
* **prototype**：一个bean的定义可以有多个实例。
* **request**：每次http请求都会创建一个bean，该作用域仅在基于web的SpringApplicationContext情形下有效。
* **session**：在一个HTTP Session中，一个bean定义对应一个实例。该作用域仅在基于web的SpringApplicationContext情形下有效。
* **global**-**session**：在一个全局的HTTP Session中，一个bean定义对应一个实例。该作用域仅在基于web的SpringApplicationContext情形下有效。

缺省的Spring bean 的作用域是Singleton.

**26. Spring框架中的单例bean是线程安全的吗?**

不，Spring框架中的单例bean不是线程安全的。

**27. 解释Spring框架中bean的生命周期。**

Spring容器从XML 文件中读取bean的定义，并实例化bean。

Spring根据bean的定义填充所有的属性。

如果bean实现了BeanNameAware 接口，Spring 传递bean 的ID 到 setBeanName方法。

如果Bean 实现了BeanFactoryAware 接口， Spring传递beanfactory 给setBeanFactory 方法。

如果有任何与bean相关联的BeanPostProcessors，Spring会在postProcesserBeforeInitialization()方法内调用它们。

如果bean实现IntializingBean了，调用它的afterPropertySet方法，如果bean声明了初始化方法，调用此初始化方法。

如果有BeanPostProcessors和bean 关联，这些bean的postProcessAfterInitialization()方法将被调用。

如果bean实现了 DisposableBean，它将调用destroy()方法。

**28. 哪些是重要的bean生命周期方法？你能重载它们吗？**

有两个重要的bean 生命周期方法，第一个是setup ，它是在容器加载bean的时候被调用。第二个方法是 teardown 它是在容器卸载类的时候被调用。

The bean 标签有两个重要的属性（init-method和destroy-method）。用它们你可以自己定制初始化和注销方法。它们也有相应的注解（@PostConstruct和@PreDestroy）。

**29. 什么是Spring的内部bean？**

当一个bean仅被用作另一个bean的属性时，它能被声明为一个内部bean，为了定义inner bean，在Spring 的基于XML的配置元数据中，可以在<property/>或 <constructor-arg/> 元素内使用<bean/> 元素，内部bean通常是匿名的，它们的Scope一般是prototype。

**30. 在 Spring中如何注入一个java集合？**

Spring提供以下几种集合的配置元素：

* <list>类型用于注入一列值，允许有相同的值。
* <set> 类型用于注入一组值，不允许有相同的值。
* <map> 类型用于注入一组键值对，键和值都可以为任意类型。
* <props>类型用于注入一组键值对，键和值都只能为String类型。

**31. 什么是bean装配?**

装配，或bean 装配是指在Spring 容器中把bean组装到一起，前提是容器需要知道bean的依赖关系，如何通过依赖注入来把它们装配到一起。

**32. 什么是bean的自动装配？**

Spring 容器能够自动装配相互合作的bean，这意味着容器不需要<constructor-arg>和<property>配置，能通过Bean工厂自动处理bean之间的协作。

**33. 解释不同方式的自动装配。**

有五种自动装配的方式，可以用来指导Spring容器用自动装配方式来进行依赖注入。

* **no：**默认的方式是不进行自动装配，通过显式设置ref 属性来进行装配。
* **byName：**通过参数名自动装配，Spring容器在配置文件中发现bean的autowire属性被设置成byname，之后容器试图匹配、装配和该bean的属性具有相同名字的bean。
* **byType:**通过参数类型自动装配，Spring容器在配置文件中发现bean的autowire属性被设置成byType，之后容器试图匹配、装配和该bean的属性具有相同类型的bean。如果有多个bean符合条件，则抛出错误。
* **constructor：**这个方式类似于byType，但是要提供给构造器参数，如果没有确定的带参数的构造器参数类型，将会抛出异常。
* **autodetect：**首先尝试使用constructor来自动装配，如果无法工作，则使用byType方式。

**34.自动装配有哪些局限性 ?**

自动装配的局限性是：

* **重写：**你仍需用<constructor-arg>和<property> 配置来定义依赖，意味着总要重写自动装配。
* **基本数据类型：**你不能自动装配简单的属性，如基本数据类型，String字符串，和类。
* **模糊特性：**自动装配不如显式装配精确，如果有可能，建议使用显式装配。

**35. 你可以在Spring中注入一个null 和一个空字符串吗？**

可以。