1.spring的控制反转IOC：inversion of control，是什么意思？

不用new对象，new的过程正向，不用new了，那怎么实例化？通过spring容器来创建

对象，以及管理对象之间的依赖关系。

2.实例化。

<bean id="xxxx" class="xxx.xxx.xxx.xxx"></bean>

id作为需要实例化类在spring容器里面的标识。id必须是唯一的，通过id可以找到该对象。

class作为实例化类的具体指定。

name，作用和id是一样的，只是命名的规则有些不一致，建议使用id来作为标识。

scope:指定实例化类的时候使用的模式。分为单例模式和多例模式。默认是singleton单例模式

，多例模式prototype

2.1 实例化bean有几种方式：构造方法、工厂方法、静态工厂方法。

1.无参构造方法。

<bean id="bean1" class="xxx.xxx.xxx.xxx"></bean>

2.有参构造方法。

<bean id="bean1" class="xxx.xxx.xxx.xxx">

<constructor-arg index="0" value="xxx"></constructor-arg>

</bean>

1和2统一起来就是通过构造方法实例化bean。

3.工厂方法。

<bean id="工厂" class="工厂全路径"></bean>

<bean id="bean" factory-bean="工厂id" factory-method="工厂中实例化对象的方法"></bean>

4.静态工厂方法。

<bean id="bean" class="静态工厂全路径" factory-method="静态工厂中实例化对象的方法"></bean>

一般使用无参构造方法实例化bean。因为无参构造方法默认会有，使用的时候最方便。

3.管理依赖关系。通过ref来引入相关的依赖类。

<bean id="bean1" class="xxx.xxx.xxx.xxx"></bean>

<bean id="bean2" class="xxx.xxx.xxx.xxx">

<property name="bean1" ref="bean1"></property>

</bean>

4. 依赖注入DI:denpendency inject

1.注入属性值 setter方法注入,必须要有该属性的setter方法

<bean id="bean1" class="xxx.xxx.xxx.xxx">

<property name="beanName" value="xxx"></property>

</bean>

2.有参构造方法

public XXX(String name){

this.name = name;

}

<bean id="bean1" class="xxx.xxx.xxx.xxx">

<constructor-arg index="0" value="xxx"></constructor-arg>

</bean>

3.注入list

<bean id="bean1" class="xxx.xxx.xxx.xxx">

<property name="xxxList">

<list>

<value>xxx</value>

<value>xxx</value>

<value>xxx</value>

<value>xxx</value>

</list>

</property>

</bean>

4.注入map

<bean id="bean1" class="xxx.xxx.xxx.xxx">

<property name="xxxMap">

<map>

<entry key="xxx" value="xxx"></entry>

<entry key="xxx" value="xxx"></entry>

<entry key="xxx" value="xxx"></entry>

<entry key="xxx" value="xxx"></entry>

</map>

</property>

</bean>

5.注入properties

<bean id="bean1" class="xxx.xxx.xxx.xxx">

<property name="xxxProperties">

<props>

<prop key="log4j.appender.stdout">org.apache.log4j.ConsoleAppender</prop>

</props>

</property>

</bean>

5.IOC相关注解。

5.1 实例化bean对象注解

@Component

@Service

@Repository

@Controller

根据功能的不同取不同的名字而已，实际作用都是实例化bean对象。

5.2 属性的注解。

1.属性是基本类型或者string类型，使用@Value

2.属性是对象类型

2.1 使用@Autowired，自动装配注解。

2.2 使用@Resource(name="xxx")

2.3 qualifier("xxxx")

6.AOP面向切面编程：Aspect orentied programming。

1.连接点：可以被增强的方法。

2.切入点：实际被增强的方法。

3.增强：实际加强的业务逻辑。

4.切面：增强运用到切入点的过程。

7.AOP的xml配置方式：

1.配置增强类

<bean id="adviceXxx" class="xxx.xxx.xxx.xxx"></bean>

<aop:config>

<aop:pointcut expression="execution(\* xxx.xxx.xxx.xxx.\*(..))" id="pointcutXXX"></aop:pointcut>

<aop:aspect ref="adviceXxx">

<aop:before method="adviceXxx里面的方法" pointcut-ref="pointcutXXX"></aop:before>

</aop:aspect>

</aop:config>

8.AOP的扫描：

开启AOP注解功能：<aop:aspectj-autoproxy></aop:aspectj-autoproxy>

增强类注解：

@Aspect注解。

增强类方法注解：

前置增强：@Before(value="execution(\* xxx.xxx.xxx.xxx.\*(..))")

后置增强：@AfterReturning(value="execution(\* xxx.xxx.xxx.xxx.\*(..))")

环绕增强：@Around(value="execution(\* xxx.xxx.xxx.xxx.\*(..))")

9.web项目的spring运用。

在web.xml文件里面配置spring容器。

1.将spring的配置文件作为web项目的参数来读取。

<context-param>

<param-name>contextConfigLocation</param-name>

<param-value>classpath:applicationContext.xml</param-value>

</context-param>

2.配置Spring的监听器。

<listener>

<listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>

</listener>

3.原理和功能：

提高spring的性能和效率。

10 log4j : log for java，通过log4j可以看到项目详细日志信息。

我提供文件，其它地方不用修改，如果是开发环境，改成debug，生产环境使用info。