SpringDay01

2016年9月19日

8:57

# Spring

主要作用:整合现在主流框架。是框架之间的调用更加的顺畅、快捷、方便管理

课程安排：

1.Spring的IOC(控制反转) 和DI（依赖注入）

2.统统使用注解形式 简化配置

3. AOP面向切面编程

4.AOP的集中练习

5.spring和JDBC的整合 Spring的声明式的事务处理

## 一、Spring的IOC

控制反转

定义：

把对象的创建初始化和销毁的工作 都交给spring容器来做。从此程序员 无需关注对象的创建过程。

1.Person p1 = new Person();//主动创建

2.Person p2 = Spring容器.get\*\*(); //对象创建的权利交给spring容器

二、1.为什么要用这种方式创建对象呢？？

2.这样写的好处？？？？

## 三、IOC的实现步骤：

1.导入相关jar包

2.创建实体类

3.编写配置文件xml(1.配置文件的头 2.配置实体类对应关系)

4.启动spring容器

5.得到相应的对象 对象调用方法 完成相应的功能。

四、具体的实现：

1.

计算机生成了可选文字:
‘仑1ib
里common、一Iogging一1·0·4jar
邑，pring一bean，一3·2.3.RELEAsEjar
气，pring一context一3,2.3.RELEASE.jar
乙spring一context一support一3.2.3.RELEA
里spring一core一3·2.3.RELEAsE.jar
乙spring一expression一3·2·3·RELEAsE.jar
里spring一test一3.2.3.RELEAsE.jar一
叼AX
人op习题

2.创建实体类

public class Hello {

//第二步 创建实体类

public void say(){

System.out.println("Hello Spring ");

}

}

## 3.配置xml文件

1.配置文件的默认的名称就是 applicationContext.xml

2.配置文件的名称任意，但是一般都写默认的

3.配置文件的头？？？

1.直接粘贴 有模板

2.通过手动生成

<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://www.springframework.org/schema/beans

<http://www.springframework.org/schema/beans/spring-beans-3.2.xsd>">

<!--只要spring容器管理对象就是一个bean

id:唯一标识 首字母小写的类名 后边驼峰规则

class:包名+类名

-->

<bean id="hello" class="domain.Hello"></bean>

</beans>

## 4.启动spring容器

//第四步：启动spring的容器

ApplicationContext context =

new ClassPathXmlApplicationContext("applicationContext.xml");

## 5.获取对象

//第五步 获取相应的对象

Hello hello = (Hello) context.getBean("hello");

hello.say();

## ：四 IOC的实现过程

当容器启动的时候首先会先加载applicationContext.xml,顺序加载

当遇到Bean标签的时候 根据class的路径找到相应的实体类并且创建对象，把创建出来的对象放到spring容器里。spring容器内部维护了一张MAP<key,value> 对象的ID就是MAP中额Key 对象就是value保存在spring器中。如果想获得相应的对象 则通过getBean（"ID"）就能得到应用的对象

计算机生成了可选文字:
口AddxMLcatalog〔lement
匡国
瓣
Location:work一3.2.3.RELEASE\schema\beanl\spring一beans一3.2.xsd
CatalogEntry
Work币蔽瓦
1lesystem…
坦
Keyfype:
Key:
Namespacename
口口口．
〕r1ngframework.org/l:c卜en、。／b4呈
spring一beans一3.2.xsd
RewriteEntry
汹
日AIternativewebaddress:
Su佑xEntry
豁
NextCatalog
氰
DelegateCatalog

计算机生成了可选文字:
sele仁tROOtElement
SeledtherootelementoftheXML石！e.
端
Rootelement:
beans
口Createoptionalattribute、
口Createoptionalelements
口Limitoptionalelementdepthto:
回Create石rstchoiceofrequiredchoice
回Fillelement、andattribute,withdata
…口
NamesoaceInformation
N
h
lespaceName
：刀～.springframework.org/schema/beans
LocationHint
http：刀～.springframework.org/:c卜ema/beans/spring一
}Add.．·｝
｛〔d;t…1
De}ete
Apre6xmustbespeci6edforthenamespacenamed:’卜ttp：刀～.springframework.org／、c卜ema/beans'
洲二入

5：常见的问题：

计算机生成了可选文字:
beans.factory.BeanDe们nitionstoreException:IOExceptionparsing
notex;st
nest〔
,rk.bean，·facto叮·xml·XmIBeanDe石nitionReader.loadBeanDe6nition,(XmIBeanD翻nitionReader.java:341)
,rk.beans.factory.xmI.XmIBeanDe6nitionReader.loadBeanDe6nitions(XmIBeanD翻nitionReaderjava:302)
,rk.beans.facto即.support.AbstractBeanDe石nitionReader.ioadBeanDe石nitions(AbstractBeanDe右nltionReader.java:174)

bean的ID编写错误

计算机生成了可选文字:
Jora．、orinafr。mework.be。n、．factory.Nosuc卜日e。noe石ni连益氮芬氛荔介花下二r品言昆节孟
=atOrg·Spr,ngTramewo『K·Deans·Tactory·suppo几·ueTau't俗留滥监留昌留留公瑞留铝昌侣留昌留监昌昌留监吕监监臼卜aCTory.Java’。O己）
三atorg.springframework.beans.facto叮．support.AbstractBeanFacto甲．getMergedLocaIBeanDe石nition(AbstractBeanFacto心．java:1108)
三atorg.springframework.beans.facto甲．support.AbstractBeanFacto叮．doGetBean(AbstractBeanFacto甲．java:275)
三atorg.springframework.beans.facto叹．,upport.AbstractBeanFacto叮．getBean(AbstractBeanFacto叮．java:194)
三atorg.springframework.context.support.Ab,tractApplicationContext.getBean(Abstra以pplicationcontextjava:1117)
三atte、t.springTe、t.test03(springTest.java:39)

6.关于bean的问题

建议使用id,因为使用class可能会造成1对二的情况

计算机生成了可选文字:
”巴r'T!"""”兰‘fy'n呈b尸月，尺，坪只望o吧‘rt'"e“尸，污翌ned严不
ln了“翌’g叮咒塑脚咒a川岁’｛翌望’子”望CT吧艺嘿：‘划L一
eXt.g巴De口n几从DStra〔毛伙ppll（口tI0n峪onteXt.J口V口：i山‘J)es
5Inglematching
e日nbUttOUn
:ne!{O
囚
111

### 7.别名标签

<!--别名标签 -->

<alias name="hello" alias="1606"/>

使用场景，需要一个class对应多个id时使用。

8.Spring对象创建的原理

//domain.Hello

Class.forName(Hello.class.getName()).newInstance();

9.无参构造的重要性

我们spring底层通过反射调用的形式创建对象，所以实体类中必须要有无参构造。以后在手动编写构造方法的时候 一定切记 要加无参构造

计算机生成了可选文字:
ouldnotinstantiatebeanclas、［domaln.Hellol:No
defaultconstructorfound;nestedexcepti
On15
java.lang.
NosuchMethodExce
:domain.Hell
．砧～一，:～。，,.1八
atOrg.Springtram糊ork.beans·tactory.Support·Abstract八utowlre仁apable匕ean卜actory·create匕eanlnstance(Abstract八utowlre仁apable匕ean卜actory.Jav之
atorg.springframework.beans.factory.support.AbstractAutowireCapableBeanFactory.doCreateBean(AbstractAutowireCapableBeanFactoryjava:487)
atorg.,pringfram,ork.bean，·factory.,upport·Ab,tractAutowireCapableBeanFactory·createBean(AbstractAutowireCapableBeanFacto甲．java:458)
atorg.springframework.beans·factory.support.AbstractBeanFacto心＄1.getobject(AbstractBeanFacto心．java:295)
.．、t．。产．
口气r勺
，护气t二产．
r吮r龟
户勺内沪勺、

# 10.spring容器创建对象的四种方式

1.通过默认的构造方法创建对象

## 2.通过静态工厂创建对象

<!--静态工厂创建对象 -->

<bean id="calendar" class="factory.StaticFactory" factory-method="getTime"></bean>

具体步骤：

当spring容器启动的时候 顺序加载xml配置文件。找到相应的bean 根据class找到静态工厂 再通过factory-method="getTime"中getTime方法创建对象，把对象放到spring容器中。

如果在静态工厂中没有static 就会

计算机生成了可选文字:
勺
rg·SprlngTrameworK·DeanS·TaCT
ameexi,tsandthatit；、static.l
石不甘eanCreationException:Errorcreatingbeanwithname'calendar,de6nedinclasspat卜·
三1atorg.springframework.bean,.fact
．、upport.ConstructorRe,olver.instantiateU,ingFactoryMethod(ConstructorRe,olver.ja、
．、upport.Ab,tract八utowireCapableBeanFacto即．in、tantiateU、ingFacto叮Met卜od(Ab、tr:
atorg.springframework.beans.factory.,upport.AbstractAutowireCapableBeanFactory.createBeanInstance(AbstractAutowire(
atorg.springframework.beans.facto甲．support.AbstractAutowireCapableBeanFacto即．doCreateBean(AbstractAutowireCapal
一一一一一一
已．
月气尸
口气，二．
一一一一一

## 

## 3.实例工厂创建对象

<!--实例工厂创建对象 -->

<bean id="newInstanceFactory" class="factory.NewInstanceFactory"></bean>

<bean id="newCalendar" factory-bean="newInstanceFactory" factory-method="getTime"></bean>

说明：

实例工厂和静态工厂最大的区别在于 工厂对象是否被创建

## 4.spring工厂创建对象

xml配置文件

<!--spring工厂创建对象 -->

<bean id="springCalendar" class="factory.SpringFactory"></bean>

需要实现一个接口 然后自动调用 getObject()

//spring工厂创建对象

public class SpringFactory implements FactoryBean<Calendar>{

@Override

public Calendar getObject() throws Exception {

return Calendar.getInstance();

}

@Override

public Class<?> getObjectType() {

// TODO Auto-generated method stub

return Calendar.class;

}

@Override

public boolean isSingleton() {

// TODO Auto-generated method stub

return false;

}

}

11.Spring容器创建对象是单例还是多例的？、

当在默认的条件下 spring容器创建对象是单例的

好处 能够提高计算机的性能，减少内存的压力。

<!--测试spring单例和多例

scope="prototype" 多例对象

默认条件下和 scope="singleton" 都是单例的

-->

<bean id="person" class="domain.Person" scope="singleton"></bean>

spring容器只负责维护单例对象， 多例对象spring只负责创建 然后自生自灭

## 12.自定义的初始化方法和销毁方法

init-method="init" destroy-method="destroy"

容器关闭

ClassPathXmlApplicationContext context =

new ClassPathXmlApplicationContext("applicationContext.xml");

Person p1 = (Person) context.getBean("person");

p1.add();

context.close();

## 13.对象的懒加载

问题：spring容器启动的时候就会根据bean来创建对象，并且是全部创建，对于这种模式程序性能不能提高

理想模式：当我需要使用某个对象的时候，这个对象才会被创建。如果我不需要的时候，那么这个对象就不会被创建出来。

<!--懒加载的配置

lazy-init="default"

lazy-init="false"         默认的就是立即加载

lazy-init="true"        进行懒加载

-->

<bean id="person" class="domain.Person" lazy-init="true"></bean>

注意：

当

<bean id="person" class="domain.Person" lazy-init="false" scope="prototype"></bean>

对象的属性scope为prototype的时候多例对象 立即加载会失效。

会变成懒加载。

注意：

lazy-init="default" 这时需要查看default-lazy-init="false"

它的匹配规则和默认的保持一致

当全局的懒加载和bean中的懒加载不一致时，按照bean中的配置执行。

关于多例对象销毁问题

当多例对象是 容器只负责创建 其他一概不管。