命名空间定义

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

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xmlns:p=*"http://www.springframework.org/schema/p"*

xmlns:context=*"http://www.springframework.org/schema/context"*

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

*http://www.springframework.org/schema/beans/spring-beans-3.0.xsd*

*http://www.springframework.org/schema/context*

*http://www.springframework.org/schema/context/spring-context-3.0.xsd"*>

<?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-2.5.xsd">

<!-- 定义使用C3P0连接池的数据源 -->

<bean id="dataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource">

<!-- 指定连接数据库的JDBC驱动 -->

<property name="driverClass">

<value>com.mysql.jdbc.Driver</value>

</property>

<!-- 连接数据库所用的URL -->

<property name="jdbcUrl">

<value>jdbc:mysql://localhost:3306/eportal?useUnicode=true&amp;characterEncoding=gbk</value>

</property>

<!-- 连接数据库的用户名 -->

<property name="user">

<value>root</value>

</property>

<!-- 连接数据库的密码 -->

<property name="password">

<value>jiangnan77</value>

</property>

<!-- 设置数据库连接池的最大连接数 -->

<property name="maxPoolSize">

<value>20</value>

</property>

<!-- 设置数据库连接池的最小连接数 -->

<property name="minPoolSize">

<value>2</value>

</property>

<!-- 设置数据库连接池的初始化连接数 -->

<property name="initialPoolSize">

<value>2</value>

</property>

<!-- 设置数据库连接池的连接的最大空闲时间,单位为秒 -->

<property name="maxIdleTime">

<value>20</value>

</property>

</bean>

//////////////////////////以上是数据库连接池/////////////////////

<!-- 定义Hibernate的SessionFactory -->

<beanid="sessionFactory"class="org.springframework.orm.hibernate3.LocalSessionFactoryBean">

<!-- 依赖注入上面定义的数据源dataSource -->

<property name="dataSource" ref="dataSource"/>///////依赖上面的datasource

<!-- 注册Hibernate的ORM映射文件 -->

<property name="mappingResources"> ////hibernate的映射文件

<list>

<value>com/eportal/ORM/News.hbm.xml</value>

<value>com/eportal/ORM/Category.hbm.xml</value>

<value>com/eportal/ORM/Memberlevel.hbm.xml</value>

<value>com/eportal/ORM/Cart.hbm.xml</value>

<value>com/eportal/ORM/Traffic.hbm.xml</value>

<value>com/eportal/ORM/Newsrule.hbm.xml</value>

<value>com/eportal/ORM/Merchandise.hbm.xml</value>

<value>com/eportal/ORM/Admin.hbm.xml</value>

<value>com/eportal/ORM/Orders.hbm.xml</value>

<value>com/eportal/ORM/Cartselectedmer.hbm.xml</value>

<value>com/eportal/ORM/Newscolumns.hbm.xml</value>

<value>com/eportal/ORM/Member.hbm.xml</value>

</list>

</property>

<!-- 设置Hibernate的相关属性 -->

<property name="hibernateProperties">

<props>

<!-- 设置Hibernate的数据库方言 -->

<prop key="hibernate.dialect">org.hibernate.dialect.MySQLDialect</prop>

<!-- 设置Hibernate是否在控制台输出SQL语句，开发调试阶段通常设为true -->

<prop key="show\_sql">true</prop>

<!-- 设置Hibernate一个提交批次中的最大SQL语句数 -->

<prop key="hibernate.jdbc.batch\_size">50</prop>

</props>

</property>

</bean>

//////////////////////hibernate映射方面的接管///////////////////////

<!--定义Hibernate的事务管理器HibernateTransactionManager --> <beanid="transactionManager"class="org.springframework.orm.hibernate3.HibernateTransactionManager">

<!-- 依赖注入上面定义的sessionFactory -->

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

</bean>

<!--定义Spring的事务拦截器TransactionInterceptor -->/////换句话说叫做切面建议 advice

<beanid="transactionInterceptor"class="org.springframework.transaction.interceptor.TransactionInterceptor">

<!-- 依赖注入上面定义的事务管理器transactionManager -->

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

<!-- 定义需要进行事务拦截的方法及所采用的事务控制类型 -->

<property name="transactionAttributes">

<props>

<!-- 以browse、list、load、get及is开头的所有方法采用只读型事务控制类型 -->

<prop key="browse\*">PROPAGATION\_REQUIRED,readOnly</prop>

<prop key="list\*">PROPAGATION\_REQUIRED,readOnly</prop>

<prop key="load\*">PROPAGATION\_REQUIRED,readOnly</prop>

<prop key="get\*">PROPAGATION\_REQUIRED,readOnly</prop>

<prop key="is\*">PROPAGATION\_REQUIRED,readOnly</prop>

<!-- 所有方法均进行事务控制，如果当前没有事务，则新建一个事务 -->

<prop key="\*">PROPAGATION\_REQUIRED</prop>

</props>

</property>

</bean>

//////////////////////////////指定什么样的方法进行事务控制/////////////

事务控制的具体实现 ----》》》》通过代理类实现 proxy

<!-- 定义BeanNameAutoProxyCreatorf进行Spring的事务处理-->

<bean class="org.springframework.aop.framework.autoproxy.BeanNameAutoProxyCreator">

<!-- 针对指定的bean自动生成业务代理 -->

<property name="beanNames"> ------》》》。制定什么类要进行代理 bean在线面定义

<list>

<value>adminService</value>

<value>columnsService</value>

<value>newsService</value>

<value>crawlService</value>

<value>memberLevelService</value>

<value>memberService</value>

<value>categoryService</value>

<value>merService</value>

<value>cartService</value>

<value>ordersService</value>

<value>trafficService</value>

</list>

</property>

<!-- 这个属性为true时，表示被代理的是目标类本身而不是目标类的接口 -->

<property name="proxyTargetClass">

<value>true</value>

</property>

<!-- 依赖注入上面定义的事务拦截器transactionInterceptor -->

<property name="interceptorNames">

<list>

<value>transactionInterceptor</value>

</list>

</property>

</bean>

<!-- 装配通用数据库访问类BaseDAOImpl -->

<bean id="dao" class="com.eportal.DAO.BaseDAOImpl">

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

</bean>

<!-- 部署系统用户管理业务逻辑组件AdminServiceImpl -->

<bean id="adminService" class="com.eportal.service.AdminServiceImpl">

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

</bean>

<!-- 部署新闻栏目管理业务逻辑组件ColumnsServiceImpl -->

<bean id="columnsService" class="com.eportal.service.ColumnsServiceImpl">

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

</bean>

<!-- 部署新闻管理业务逻辑组件NewsServiceImpl -->

<bean id="newsService" class="com.eportal.service.NewsServiceImpl">

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

</bean>

<!-- 部署新闻采集规则业务逻辑组件CrawlServiceImpl -->

<bean id="crawlService" class="com.eportal.service.CrawlServiceImpl">

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

</bean>

<!-- 部署会员级别管理业务逻辑组件MemberLevelServiceImpl -->

<bean id="memberLevelService" class="com.eportal.service.MemberLevelServiceImpl">

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

</bean>

<!-- 部署会员管理业务逻辑组件MemberServiceImpl -->

<bean id="memberService" class="com.eportal.service.MemberServiceImpl">

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

</bean>

<!-- 部署商品分类管理业务逻辑组件CategoryServiceImpl -->

<bean id="categoryService" class="com.eportal.service.CategoryServiceImpl">

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

</bean>

<!-- 部署商品管理业务逻辑组件MerchandiseServiceImpl -->

<bean id="merService" class="com.eportal.service.MerchandiseServiceImpl">

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

</bean>

<!-- 部署购物车管理业务逻辑组件CartServiceImpl -->

<bean id="cartService" class="com.eportal.service.CartServiceImpl">

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

</bean>

<!-- 部署订单管理业务逻辑组件OrderServiceImpl -->

<bean id="ordersService" class="com.eportal.service.OrderServiceImpl">

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

</bean>

<!-- 部署流量统计业务逻辑组件TrafficServiceImpl -->

<bean id="trafficService" class="com.eportal.service.TrafficServiceImpl">

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

</bean>

<!-- 部署Struts2负责系统用户管理的控制器AdminAction -->

<bean id="adminAction" class="com.eportal.struts.action.AdminAction" scope="prototype">

<property name="service" ref="adminService"/>

</bean>

<!-- 部署Struts2负责新闻栏目管理的控制器ColumnsAction -->

<bean id="columnsAction" class="com.eportal.struts.action.ColumnsAction" scope="prototype">

<property name="service" ref="columnsService"/>

</bean>

<!-- 部署Struts2负责新闻管理的控制器NewsAction -->

<bean id="newsAction" class="com.eportal.struts.action.NewsAction" scope="prototype">

<property name="service" ref="newsService"/>

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

</bean>

<!-- 部署Struts2负责新闻采集规则管理的控制器CrawlAction -->

<bean id="crawlAction" class="com.eportal.struts.action.CrawlAction" scope="prototype">

<property name="service" ref="crawlService"/>

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

</bean>

<!-- 部署Struts2负责会员级别管理的控制器MemberlevelAction -->

<bean id="memberLeveAction" class="com.eportal.struts.action.MemberlevelAction" scope="prototype">

<property name="service" ref="memberLevelService"/>

</bean>

<!-- 部署Struts2负责会员管理的控制器MemberAction -->

<bean id="memberAction" class="com.eportal.struts.action.MemberAction" scope="prototype">

<property name="service" ref="memberService"/>

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

</bean>

<!-- 部署Struts2负责商品分类管理的控制器CategoryAction -->

<bean id="categoryAction" class="com.eportal.struts.action.CategoryAction" scope="prototype">

<property name="service" ref="categoryService"/>

</bean>

<!-- 部署Struts2负责商品管理的控制器MerchandiseAction -->

<bean id="merchandiseAction" class="com.eportal.struts.action.MerchandiseAction" scope="prototype">

<property name="service" ref="merService"/>

<property name="cateService" ref="categoryService"/>

</bean>

<!-- 部署Struts2负责购物车管理的控制器CartAction -->

<bean id="cartAction" class="com.eportal.struts.action.CartAction" scope="prototype">

<property name="service" ref="cartService"/>

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

</bean>

<!-- 部署Struts2负责订单管理的控制器OrdersAction -->

<bean id="ordersAction" class="com.eportal.struts.action.OrdersAction" scope="prototype">

<property name="service" ref="ordersService"/>

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

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

</bean>

<!-- 部署Struts2负责流量统计的控制器TrafficAction -->

<bean id="trafficAction" class="com.eportal.struts.action.TrafficAction" scope="prototype">

<property name="service" ref="trafficService"/>

</bean>

</beans>