1. 导入架包

commons-beanutils、commons-collections4、commons-dbcp、commons-digester、commons-pool2、log4j、mybatis、mybatis-spring、ojdbc7、junit、slf4j-api、slf4j-log4j12、jcl-over-slf4j、servlet-api、spring-core、spring-beans、spring-context、spring-context-support、spring-orm、spring-jdbc、spring-tx、spring-web、spring-webmvc、spring-aop、spring-aspects、tiles-jsp、tiles-api、tiles-template、jstl、jackson-mapper-asl、jackson-core-asl、fastjson

1. web.xml

1、先定义过滤器，把所有请求转换编码为UTF-8

<filter>

<filter-name>characterEncodingFilter</filter-name>

<filter-class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>

<init-param>

<param-name>encoding</param-name>

<param-value>UTF-8</param-value>

</init-param>

</filter>

<filter-mapping>

<filter-name>characterEncodingFilter</filter-name>

<url-pattern>/\*</url-pattern>

</filter-mapping>

2、初始页面和默认出错页面

<welcome-file-list>

<welcome-file>login.jsp</welcome-file>

</welcome-file-list>

<error-page>

<exception-type>java.lang.Throwable</exception-type>

<location>/view/error.jsp</location>

</error-page>

3、建立消息分发器，web-servlet.xml进行视图解析

<servlet>

<servlet-name>dispatcher</servlet-name>

<servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>

<init-param>

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

<param-value>/WEB-INF/web-servlet.xml</param-value>

</init-param>

<!-- 随Servlet容器一起启动 -->

<load-on-startup>1</load-on-startup>

</servlet>

<servlet-mapping>

<servlet-name>dispatcher</servlet-name>

<url-pattern>\*.do</url-pattern>

</servlet-mapping>

4、建立监听器，applicationContext.xml配置应用上下文

<listener>

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

</listener>

<context-param>

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

<param-value>/WEB-INF/applicationContext.xml</param-value>

</context-param>

5、自定义拦截器，拦截所有请求

<filter>

<filter-name>requestFilter</filter-name>

<filter-class>com.loan.pms.system.web.filter.RequestFilter</filter-class>

<init-param>

<param-name>exclude</param-name>

<param-value>/login.do</param-value>

</init-param>

<init-param>

<description>allow request method</description>

<param-name>methods</param-name>

<param-value>GET,POST,PUT</param-value>

</init-param>

</filter>

<!-- 拦截所有请求 -->

<filter-mapping>

<filter-name>requestFilter</filter-name>

<url-pattern>/\*</url-pattern>

</filter-mapping>

1. web-servlet.xml

1、扫描所有bean

<!-- 启用 spring mvc 注解 -->

<mvc:annotation-driven/>

<!-- 对com.yue下所有包下的类的注解进行扫描,并自动创建bean实例和装配bean -->

<context:component-scan base-package="com.loan.pms"></context:component-scan>

2、配置tiles视图框架，tiles-defs.xml定义页面初始布局

<!-- 优先匹配tiles视图 -->

<bean id="tilesViewResolver" class="org.springframework.web.servlet.view.UrlBasedViewResolver">

<property name="viewClass">

<value>

org.springframework.web.servlet.view.tiles2.TilesView

</value>

</property>

<property name="order" value="1" />

</bean>

<bean id="tilesConfigurer" class="org.springframework.web.servlet.view.tiles2.TilesConfigurer">

<property name="definitions">

<list>

<value>/WEB-INF/config/tiles-defs.xml</value>

</list>

</property>

</bean>

3、配置视图解析器，加上开头和结尾

<!-- 配置视图解析器 -->

<bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">

<!-- 前缀 -->

<!--<property name="prefix" value="/WEB-INF/view/"></property> -->

<!-- 后缀 -->

<property name="suffix" value=".jsp"></property>

<property name="order" value="2" />

</bean>

1. applicationContext.xml

1、引入数据库连接配置文件，jdbc.properties

<!-- 引入配置文件 -->

<bean id="propertyConfigurer"

class="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer">

<property name="location" value="/WEB-INF/config/jdbc.properties" />

</bean>

2、建立数据库连接池

<!-- 数据源配置, 使用应用中的DBCP数据库连接池 -->

<bean id="dataSource" class="org.apache.commons.dbcp.BasicDataSource"

destroy-method="close">

<!-- Connection Info -->

<property name="driverClassName" value="${jdbc.driver}" />

<property name="url" value="${jdbc.url}" />

<property name="username" value="${jdbc.username}" />

<property name="password" value="${jdbc.password}" />

<!-- Connection Pooling Info -->

<property name="maxActive" value="${dbcp.maxActive}" />

<property name="maxIdle" value="${dbcp.maxIdle}" />

<property name="defaultAutoCommit" value="false" />

<!-- 连接Idle一个小时后超时 -->

<property name="timeBetweenEvictionRunsMillis" value="3600000" />

<property name="minEvictableIdleTimeMillis" value="3600000" />

</bean>

3、spring整合mybatis，对于sql文件配置在mybatis-config.xml

<!-- spring和MyBatis完美整合，不需要mybatis的配置映射文件 -->

<bean id="sqlSessionFactory"

class="org.mybatis.spring.SqlSessionFactoryBean">

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

<!-- 自动扫描mapping.xml文件 -->

<property name="configLocation" value="/WEB-INF/config/mybatis-config.xml"></property>

</bean>

<!-- DAO接口所在包名，Spring会自动查找其下的类 -->

<bean class="org.mybatis.spring.mapper.MapperScannerConfigurer">

<property name="basePackage" value="com.loan.pms.\*.dao"></property>

<property name="sqlSessionFactoryBeanName" value="sqlSessionFactory"></property>

</bean>

4、事物回滚，回滚情形

<!-- (事务管理)transaction manager, use JtaTransactionManager for global tx -->

<bean id="transactionManager"

class="org.springframework.jdbc.datasource.DataSourceTransactionManager">

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

</bean>

1. jdbc.properties

数据库连接配置：

#连接数据库

jdbc.driver=oracle.jdbc.OracleDriver

jdbc.url=jdbc:oracle:thin:@localhost:1521:PMS

jdbc.username=PMSOPR

jdbc.password=Pms12345

#dbcp settings

dbcp.maxIdle=5

dbcp.maxActive=50

1. mybatis-config.xml

映射所有sql语句所在路径：

<configuration>

<mappers>

<mapper resource="com/loan/pms/system/dao/sql/system.xml"></mapper>

<mapper resource="com/loan/pms/product/dao/sql/product.xml"></mapper>

</mappers>

</configuration>

1. tiles-defs.xml

页面布局：

<!-- 主布局 -->

<definition name="mainLayout" template="/WEB-INF/view/mainLayout.jsp" >

<put-attribute name="title" value="信贷产品管理系统" />

<put-attribute name="north" value="/layout/head.jsp" />

<put-attribute name="west" value="/layout /menu.jsp" />

<put-attribute name="center" value="/layout /menu.jsp" />

<put-attribute name="south" value="/layout /foot.jsp" />

</definition>

<definition name="indexView" extends="mainLayout" >

<put-attribute name="title" value="主页面" />

<put-attribute name="center" value="/layout /index.jsp" />

</definition>

1. log4j.properties

配置自动匹配log4j日志

#定义LOG输出级别

log4j.rootLogger=INFO,Console,File

#定义日志输出目的地为控制台

log4j.appender.Console=org.apache.log4j.ConsoleAppender

log4j.appender.Console.Target=System.out

#可以灵活地指定日志输出格式,下面一行是指定具体的格式

log4j.appender.Console.layout=org.apache.log4j.PatternLayout

log4j.appender.Console.layout.ConversionPattern=[%c] - %m%n

#文件大小达到指定尺寸的时候产生一个新的文件

log4j.appender.File=org.apache.log4j.RollingFileAppender

#指定输出目录

log4j.appender.File.File=logs/ssm.log

#定义文件最大大小

log4j.appender.File.MaxFileSize=10MB

#输出所有日志,如果换成DEBUG表示输出DEBUG以上级别日志

log4j.appender.File.Threshold=ALL

log4j.appender.File.layout=org.apache.log4j.PatternLayout

log4j.appender.File.layout.ConversionPattern =[%p] [%d{yyyy-MM-dd HH\:mm\:ss}][%c]%m%n

#mybatis

log4j.logger.java.sql.Connection=DEBUG

log4j.logger.java.sql.PreparedStatement=DEBUG

log4j.logger.java.sql.ResultSet=DEBUG