基础项目搭建

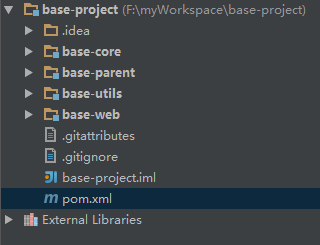
项目名称：base-project

相关技术：spring-3.1.2.RELEASE，hibernate-3.5.6-Final，mysql，maven， freemarker

该项目使用springMVC+Hibernate框架，利用maven进行架构，mysql进行数据存储，前端用freemarker进行展示。

# 建立项目基本结构

项目结构如下图：



该项目是在intellij idea中建立的，其中.idea, .gitattributes, .gitignore, base-project.iml文件是intellij自动生成的，与本项目无关。

## base-project

该项目使用了maven的聚合与继承，base-project是项目最外层，聚合了base-parent、base-core、base-utils、 base-web。

base-project的pom文件如下：

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.base</groupId>

<artifactId>base-project</artifactId>

<version>${base.version}</version>

<packaging>pom</packaging>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<base.version>1.0-SNAPSHOT</base.version>

</properties>

<modules>

<module>base-parent</module>

<module>base-utils</module>

<module>base-core</module>

<module>base-web</module>

</modules>

</project>

## base-parent

该项目使用了maven的聚合与继承，base-project聚合了base-parent、base-core、base-utils、 base-web，而base-parent为base-core，base-utils，base-web的父项目，base-core，base-utils，base-web继承了base-parent，继承关系在各自项目的pom文件中有体现，将在后面一一介绍。

base-project 的pom文件如下：

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

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4\_0\_0.xsd">

<groupId>${project.groupId}</groupId>

<artifactId>base-parent</artifactId>

<version>${base.version}</version>

<modelVersion>4.0.0</modelVersion>

<packaging>pom</packaging>

<name>base-parent</name>

<properties>

<junit.version>4.10</junit.version>

<hibernate.version>3.5.6-Final</hibernate.version>

<spring.version>3.1.2.RELEASE</spring.version>

</properties>

<dependencyManagement>

<dependencies>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>${junit.version}</version>

<scope>test</scope>

</dependency>

</dependencyManagement>

</project>

后续将会在该pom文件中加入包的依赖

## base-utils

base-utils为项目提供工具类，将会打成jar包被base-core或其他项目引用。

base-utils的pom文件如下：

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

<project xmlns="http://maven.apache.org/POM/4.0.0"

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

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<parent>

<artifactId>base-parent</artifactId>

<groupId>com.base</groupId>

<version>${base.version}</version>

<relativePath>../base-parent/pom.xml</relativePath>

</parent>

<name>base-utils</name>

<modelVersion>4.0.0</modelVersion>

<artifactId>base-utils</artifactId>

</project>

该pom文件继承了base-parent的pom文件，包的引用将在后续进行添加。

## base-core

base-core为项目核心，将打成jar包被base-web引用。base-core中主要包含项目的配置文件（数据库配置），领域对象，dao层，service层。该项目还引用了base-utils打成的jar包

base-core的pom文件如下：

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

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4\_0\_0.xsd">

<modelVersion>4.0.0</modelVersion>

<parent>

<groupId>com.base</groupId>

<artifactId>base-parent</artifactId>

<version>${base.version}</version>

<relativePath>../base-parent/pom.xml</relativePath>

</parent>

<packaging>jar</packaging>

<name>base-core</name>

<artifactId>base-core</artifactId>

<dependencies>

<dependency>

<groupId>${project.groupId}</groupId>

<artifactId>base-utils</artifactId>

<version>1.0-SNAPSHOT</version>

</dependency>

</dependencies>

</project>

## base-web

base-web为web层，是一个web项目，引用了base-core。

主要包含controller层，静态资源，展示页面（freemarker）。

base-web的pom文件如下：

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

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4\_0\_0.xsd">

<parent>

<artifactId>base-parent</artifactId>

<groupId>com.base</groupId>

<version>${base.version}</version>

<relativePath>../base-parent/pom.xml</relativePath>

</parent>

<modelVersion>4.0.0</modelVersion>

<packaging>war</packaging>

<name>base-web</name>

<artifactId>base-web</artifactId>

<dependencies>

<dependency>

<groupId>${project.groupId}</groupId>

<artifactId>base-core</artifactId>

<version>1.0-SNAPSHOT</version>

</dependency>

</dependencies>

</project>

至此，项目基本结构建成。接下来引入spring和hibernate

# 引入spring和hibernate

1. **引入spring和springMVC的依赖包**

在base-parent的pom文件中加入如下配置：

<dependencyManagement>

<dependencies>

<!--spring start-->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-core</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-orm</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-web</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-tx</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-test</artifactId>

<version>${spring.version}</version>

<scope>test</scope>

</dependency>

<!--spring end-->

</dependencies>

</dependencyManagement>

然后在base-web的pom文件中加入以上配置（不需要版本号）

由于在base-web的pom文件继承了base-parent的pom，所以在此引用时不需要<version></version>，以后的配置都遵循该方式

1. **引入hibernate依赖包**

base-parent的pom文件中加入如下配置：

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-core</artifactId>

<version>${hibernate.version}</version>

</dependency>

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-annotations</artifactId>

<version>${hibernate.version}</version>

</dependency>

<dependency>

<groupId>commons-dbcp</groupId>

<artifactId>commons-dbcp</artifactId>

<version>1.4</version>

</dependency>

然后在base-core的pom文件中加入以上配置（不需要版本号）：

1. **配置数据库连接**

加入数据库相关jar包

在base-parent的pom中加入：

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>5.1.27</version>

</dependency>

在base-core的pom中加入：

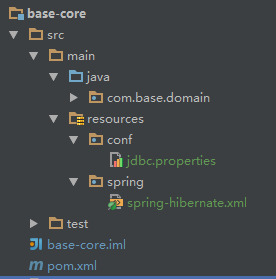
<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

</dependency>

然后在base-core项目的resources文件夹下建立两个文件夹，分别为conf和spring，然后再conf下建立jdbc.properties。在spring文件夹下建立spring-hibernate.xml，目录结构如下图：



**Jdbc.properities配置如下：**

jdbc.driverClassName=com.mysql.jdbc.Driver

jdbc.url=jdbc:mysql://localhost:3308/my-project

jdbc.username=root

jdbc.password=admin

**spring-hibernate.xml配置如下：**

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

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

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

xmlns:tx="http://www.springframework.org/schema/tx"

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

xsi:schemaLocation="

http://www.springframework.org/schema/beans

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

http://www.springframework.org/schema/tx

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

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

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

default-autowire="byName" default-lazy-init="false">

<!--载入jdbc配置文件 -->

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

<property name="locations">

<value>classpath:conf/jdbc.properties</value>

</property>

</bean>

<bean id="dataSource" class="org.apache.commons.dbcp.BasicDataSource" destroy-method="close">

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

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

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

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

</bean>

<bean id="sessionFactory" class="org.springframework.orm.hibernate3.annotation.AnnotationSessionFactoryBean">

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

<property name="hibernateProperties">

<props>

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

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

<prop key="hibernate.format\_sql">true</prop>

<prop key="hibernate.hbm2ddl.auto">update</prop>

</props>

</property>

<!--扫描所有的领域对象-->

<property name="packagesToScan" value="com.base.domain.\*"></property>

</bean>

<context:annotation-config />

<context:component-scan base-package="com.base.\*"/>

<bean id="transactionManager" class="org.springframework.orm.hibernate3.HibernateTransactionManager" />

<tx:annotation-driven transaction-manager="transactionManager"/>

</beans>

至此，数据库连接配置完成，接下配置springMVC和web.xml

1. **配置springMVC**
2. 在base-web的pom文件中引入如下配置：

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-web</artifactId>

</dependency>

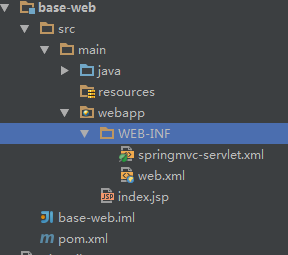
<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

</dependency>

1. 在base-web项目的WEB-INFO下创建springmvc-servlet.xml文件，如下图：



springmvc-servlet.xml的内容为：

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

<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.xsd

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

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

default-autowire="byName" default-lazy-init="false">

<import resource="classpath\*:/spring/spring-\*.xml"/>

<context:component-scan base-package="com.base\*"/>

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

<property name="viewClass" value="org.springframework.web.servlet.view.JstlView" />

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

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

</bean>

</beans>

1. 在web.xml添加如下配置：

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

<web-app version="2.4"

xmlns="http://java.sun.com/xml/ns/j2ee"

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

xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee http://java.sun.com/xml/ns/j2ee/web-app\_2\_4.xsd">

<welcome-file-list>

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

</welcome-file-list>

<context-param>

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

<param-value>

classpath\*:/spring/spring-\*.xml

</param-value>

</context-param>

<listener>

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

</listener>

<servlet>

<servlet-name>springmvc</servlet-name>

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

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

</servlet>

<servlet-mapping>

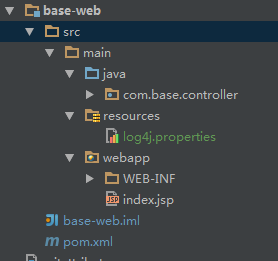
<servlet-name>springmvc</servlet-name>

<url-pattern>/</url-pattern>

</servlet-mapping>

</web-app>

1. 配置log4j
2. 在base-web项目下的resources文件夹下加入log4j.properties文件，目录结构如下：



Log4j.properites配置内容如下：

log4j.rootLogger=INFO, A1

log4j.appender.A1=org.apache.log4j.ConsoleAppender

log4j.appender.A1.layout=org.apache.log4j.PatternLayout

log4j.appender.A1.layout.ConversionPattern=%-4r %-5p [%t] %37c %3x - %m%n

### 输出到日志文件 ###

#log4j.appender.D = org.apache.log4j.DailyRollingFileAppender

#log4j.appender.D.File = logs/log.log

#log4j.appender.D.Append = true

#log4j.appender.D.Threshold = DEBUG ## 输出DEBUG级别以上的日志

#log4j.appender.D.layout = org.apache.log4j.PatternLayout

#log4j.appender.D.layout.ConversionPattern = %-d{yyyy-MM-dd HH:mm:ss} [ %t:%r ] - [ %p ] %m%n

### 保存异常信息到单独文件 ###

#log4j.appender.E = org.apache.log4j.DailyRollingFileAppender

#log4j.appender.E.File = /logs/error.log ## 异常日志文件名

#log4j.appender.E.Append = true

#log4j.appender.E.Threshold = ERROR ## 只输出ERROR级别以上的日志!!!

#log4j.appender.E.layout = org.apache.log4j.PatternLayout

#log4j.appender.E.layout.ConversionPattern = %-d{yyyy-MM-dd HH:mm:ss} [ %l:%c:%t:%r ] - [ %p ] %m%n

1. 在base-parent的pom文件中引入如下配置：

<dependency>

<groupId>commons-logging</groupId>

<artifactId>commons-logging</artifactId>

<version>1.1.2</version>

</dependency>

<dependency>

<groupId>log4j</groupId>

<artifactId>log4j</artifactId>

<version>1.2.17</version>

</dependency>

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-log4j12</artifactId>

<version>1.6.1</version>

</dependency>

然后在base-core的pom文件中加入以上配置（不需要版本号）

1. 在web.xml中加入如下配置：

<context-param>

<param-name>log4jConfigLocation</param-name>

<param-value>classpath:log4j.properties</param-value>

</context-param>

<listener>

<listener-class>org.springframework.web.util.Log4jConfigListener</listener-class>

</listener>

该配置必须在

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

1. Junit测试配置
2. 在base-parent的pom文件中引入如下配置：

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>${junit.version}</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-test</artifactId>

<version>${spring.version}</version>

<scope>test</scope>

</dependency>

然后在base-core的pom文件中引入以上配置（不需要版本号）

1. 在base-core项目下test🡪java文件夹中创建SpringBaseTest.java，其内容如下：

@ContextConfiguration(locations={"classpath:/spring/\*.xml"})

@TransactionConfiguration(defaultRollback=false)

@RunWith(SpringJUnit4ClassRunner.class)

@Transactional

public class SpringBaseTest extends

AbstractTransactionalJUnit4SpringContextTests {

}

测试类代码：

public class UserTest extends SpringBaseTest{

@Resource

private UserService userService;

@Test

public void userTest(){

User user = userService.get(1);

System.out.println(user.getName());

}

}

至此，基础项目基本搭建完成。