整合步骤：

1、创建web项目

2、导入mybatis、spring、springmvc、数据源、驱动的所有jar包

log4j-1.2.17.jar

log4j-api-2.0-rc1.jar

log4j-core-2.0-rc1.jar

mybatis-3.2.7.jar　　mybatis核心包

mybatis-spring-1.2.2.jar　　mybatis 与 spring整合包

mysql-connector-java-5.1.7-bin.jar　　mysql驱动包

slf4j-api-1.7.5.jar

slf4j-log4j12-1.7.5.jar

spring-aop-4.2.4.RELEASE.jar

spring-aspects-4.2.4.RELEASE.jar

spring-beans-4.2.4.RELEASE.jar

spring-context-4.2.4.RELEASE.jar

spring-context-support-4.2.4.RELEASE.jar

spring-core-4.2.4.RELEASE.jar

spring-expression-4.2.4.RELEASE.jar

spring-jdbc-4.2.4.RELEASE.jar

spring-orm-4.2.4.RELEASE.jar

spring-tx-4.2.4.RELEASE.jar

spring-web-4.2.4.RELEASE.jar

spring-webmvc-4.2.4.RELEASE.jar

asm-3.3.1.jar

cglib-2.2.2.jar

com.springsource.org.aopalliance-1.0.0.jar

com.springsource.org.apache.commons.logging-1.1.1.jar

com.springsource.org.aspectj.weaver-1.6.8.RELEASE.jar

commons-logging-1.1.1.jar

druid-0.2.20.jar　　阿里的数据源包 相当于c3p0

javassist-3.17.1-GA.jar

jstl-1.2.jar

3、配置web.xml

配置编码过滤器

```
<!-- 配置编码过滤器：编码过滤器只能对post请求起作用，编码过滤器配置在第一个位置，参数必须先编码 -->
 <filter>
  <filter-name>characterEncoding</filter-name>
  <filter-class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>
  <!-- 指定初始化编码 -->
  <init-param>
   <param-name>encoding</param-name>
   <param-value>utf-8</param-value>
  </init-param>
 </filter>
 <filter-mapping>
  <filter-name>characterEncoding</filter-name>
  <!-- 拦截所有请求，对所有请求参数都需要编码 -->
  <url-pattern>/*</url-pattern>
 </filter-mapping>
```

加载**Spring**配置

```
<!--监听加载Spring配置文件-->
 <listener>
  <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
```

```
</listener>
<context-param>
 <param-name>contextConfigLocation</param-name>
 <param-value>classpath:applicationContext.xml</param-value>
</context-param>
```

加载**Springmvc**配置文件

```
<!-- 前端控制器    web项目：入口配置文件web.xml
 1、加载spring配置文件      2、加载Springmvc配置文件    -->
<servlet>
 <servlet-name>springmvc</servlet-name>
 <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
 <!--
 springmvc默认加载springmvc配置文件： 默认加载springmvc配置文件必须满足约定：
 文件名称：servlet-name-servlet.xml springmvc-servlet.xml    文件路径：必须在WEB-INF下面  -->
 <!-- 自定义加载springmvc配置文件 -->
 <init-param>
  <param-name>contextConfigLocation</param-name>
  <param-value>classpath:springmvc.xml</param-value>
 </init-param>
</servlet>
<servlet-mapping>
 <servlet-name>springmvc</servlet-name>
 <url-pattern>*.do</url-pattern>
</servlet-mapping>
```

4、创建mybatis、spring、springmvc配置文件

applicationContext.xml spring核心配置文件

配置：

1、service给spring管理

```
<!-- service给spring管理 -->
<context:component-scan base-package="com.itheima.service">
</context:component-scan>
```

2、配置数据源，连接数据库

```
<!-- 配置数据源，连接数据库 -->
<context:property-placeholder location="classpath:jdbc.properties"/>
<bean id="dataSource" class="com.alibaba.druid.pool.DruidDataSource">
<property name="url" value="${jdbc.url}"></property>
<property name="driverClassName" value="${jdbc.driver}"></property>
<property name="username" value="${jdbc.username}"></property>
<property name="password" value="${jdbc.password}"></property>
</bean>
```

3、mybatis交给spring管理

```
<!-- 第二：配置工厂.生产sqlSession -->
<bean id="sqlSessionFactory" class="org.mybatis.spring.SqlSessionFactoryBean">
<property name="dataSource" ref="dataSource"></property>
<!-- 配置别名 -->
<property name="typeAliasesPackage" value="com.itheima.domain"></property>
<!-- 加载Mybatis核心配置文件 -->
<property name="configLocation" value="classpath:sqlMapConfig.xml"></property>
</bean>
```

4、生成所有数据持久层代理对象给Spring管理

```
<!-- 第三：接口代理开发 -->
<!-- 生成所有数据持久层代理对象 -->
<bean class="org.mybatis.spring.mapper.MapperScannerConfigurer">
<property name="basePackage" value="com.itheima.mapper"></property>
</bean>
```

5、spring aop管理事务

```
<!-- 第四：spring aop管理事务，定义spring事务管理平台 -->
 <bean id="transactionManager"
class="org.springframework.jdbc.datasource.DataSourceTransactionManager">
    <property name="dataSource" ref="dataSource"></property>
 </bean>
 <!-- 事务通知 -->
  <tx:advice id="txAdvice" transaction-manager="transactionManager">
<tx:attributes>
  <tx:method name="save*" propagation="REQUIRED"/>
  <tx:method name="insert*" propagation="REQUIRED"/>
  <tx:method name="delete*" propagation="REQUIRED"/>
  <tx:method name="update*" propagation="REQUIRED"/>
  <tx:method name="get*" propagation="REQUIRED"/>
  <tx:method name="*" propagation="REQUIRED"/>
  </tx:attributes>
</tx:advice>
<!-- 事务切面 -->
  <aop:config>
    <aop:advisor advice-ref="txAdvice" pointcut="execution(* com.itheima.service.*.*
(..))"/>
  </aop:config>
```

jdbc.properties

log4j.properties

springmvc.xml　　springmvc核心配置文件

1、把Controller交给springmvc管理

```
<!-- 把Controller交给springmvc管理 -->
 <context:component-scan base-package="com.itheima.controller"></context:component-
```

scan>

2、配置注解驱动

<!-- 配置注解驱动

默认创建：处理器映射器，处理器适配器，自动提供springmvc对json格式数据支持 -->

<mvc:annotation-driven/>

3、配置视图解析器

<!-- 配置视图解析器

功能：解析真正的物理视图

解析方案：前缀+逻辑视图+后缀 组合成物理视图 /WEB-INF/jsps/hello.jsp-->

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

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

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

</bean>

sqlMapConfig.xml　　mybatis核心配置文件

5、逆向工程生成javaBean、映射文件、接口

6、创建三层架构

com.rong.controller　控制层

com.rong.domain　javaBean

com.rong.mapper　dao数据持久层

com.rong.service　服务层（业务处理）

7、使用

1、service层通过注解声明

```
@Service    通过注解创建service对象，不再去配置
public class ItemsServiceImpl implements ItemsService {

    //注入mapper接口代理对象
    @Autowired    通过注解注入数据持久层对象
    private ItemsMapper itemsMapper;
```

2、controller层通过注解声明

```java
@Controller
public class ItemsController {

    //注入Service接口代理对象
    @Autowired
    private ItemsService itemsService;

    /**
     * 需求：查询所有商品
     */
    @RequestMapping("itemList")
    public String itemList(Model model){
        //查询
        List<Items> itemsList = itemsService.findItemsList();
        //页面回显
        model.addAttribute("itemsList", itemsList);

        return "itemsList";
    }
```

生成

注入