

SSM整合

Spring整合其他几个

- SpringMVC
- MyBatis

SpringIOC容器管理：

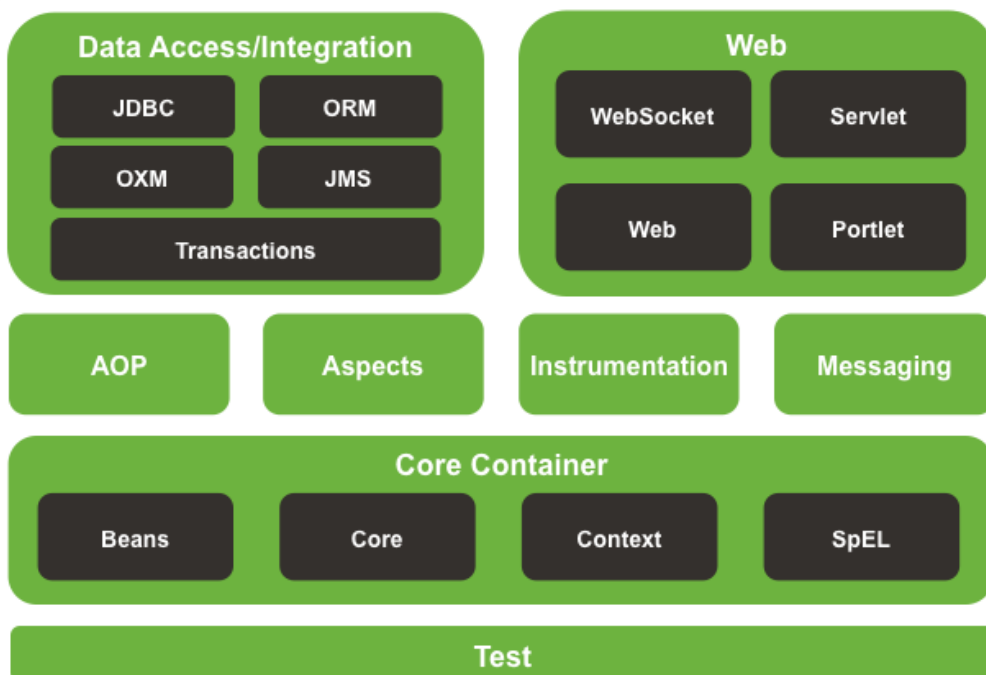
- service
- dao

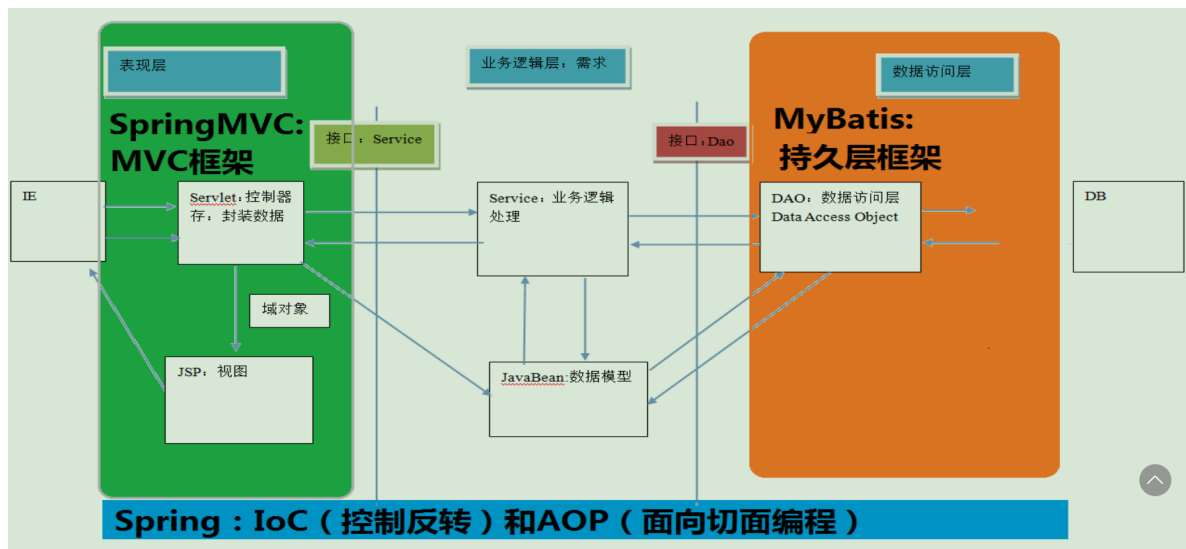
SpringMVC：

- controller



Spring Framework Runtime



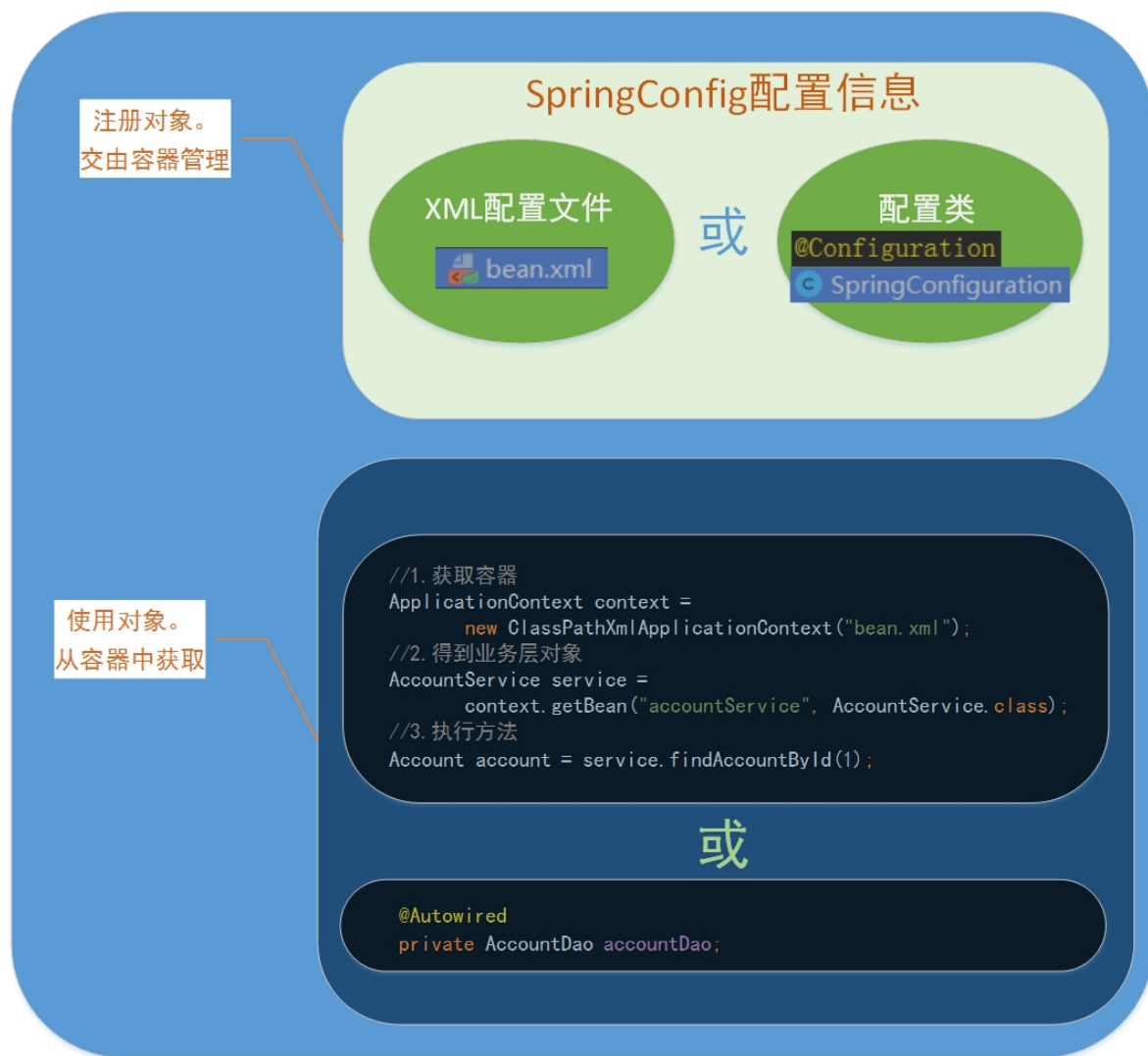


Spring需要

- service
- dao

依赖

- aspectjweaver
- spring-aop
- spring-context



配置文件 | 配置类

- 配置文件 | 配置类
 - 注册对象，交由容器管理

获取&使用 对象

- 从容器中获取对象，使用

SpringMVC需要

依赖

- spring-web

- spring-webmvc
- servlet-api
- jsp-api

- jstl

web.xml

- web.xml
 - 前端控制器
 - 加载SpringMVC.xml配置文件
 - 编码过滤器

SpringMVC.xml

- SpringMVC.xml(和Spring的配置文件是一样的)
 - 扫描包
 - 视图解析器
 - mvc注解支持
 - 类型转换
 - 静态资源不过滤
 - 拦截器注册
 - 文件解析器
 - 异常处理器

Controller

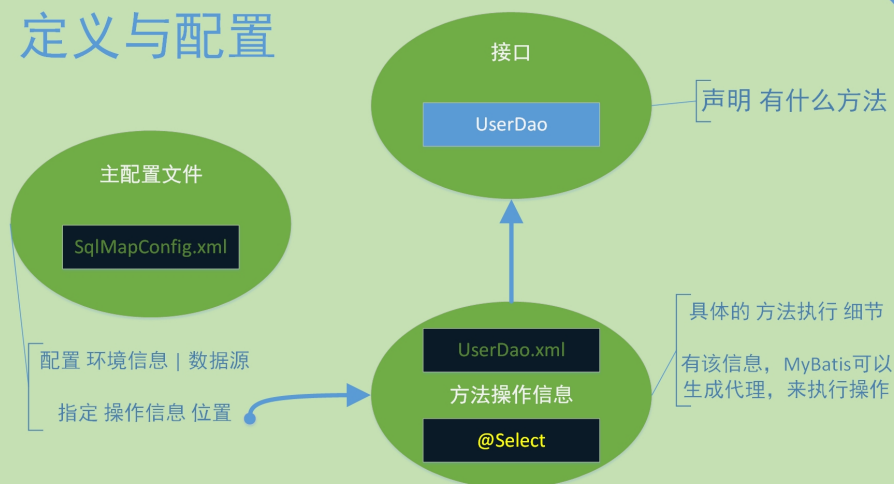
- Controller.java
 - @RequestMapping
 - @RequestParam
 - @RequestBody
 - @PathVariable
 - @RequestHeader
 - @CookieValue
 - @ModelAttribute
 - @SessionAttribute
 - @ResponseBody

MyBatis需要

依赖

- mysql-connector-java
- mybatis

定义与配置



```
//1. 读取配置文件
InputStream in = Resources.getResourceAsStream("SqlMapConfig.xml");
//2. 创建SqlSessionFactory工厂
SqlSessionFactoryBuilder builder = new SqlSessionFactoryBuilder();
SqlSessionFactory factory = builder.build(in);
//3. 使用工厂生产SqlSession对象
SqlSession session = factory.openSession();
//4. 使用SqlSession创建Dao接口的代理对象
UserDao dao = session.getMapper(UserDao.class);
//5. 使用代理对象执行方法
List<User> users = dao.findAll();
for (User user : users) {
    System.out.println(user);
}
//6. 释放资源
session.close();
in.close();
```

测试类
使用MyBatis

- 主配置文件
- 接口：声明方法
- xml|注解：方法操作信息
- 测试类使用

主配置文件SqlMapConfig.xml

- 习惯上 `SqlMapConfig.xml`
- 数据源配置
- 指定映射信息位置
- 实体类别名

接口

- 声明方法

映射配置

- xml文件 或 接口注解
- SQL执行 相关信息

获取session&使用

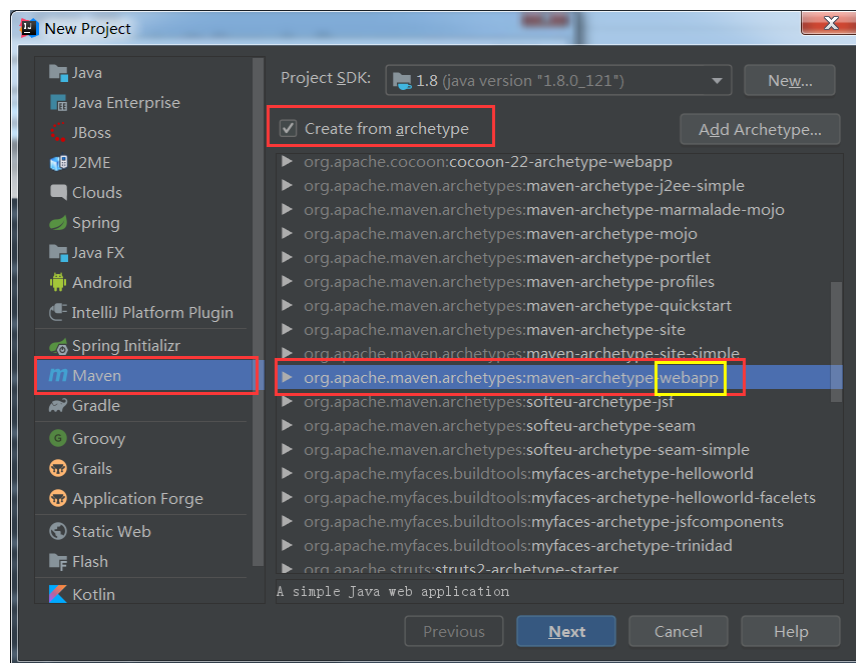
- 读取配置主配置文件，in
- Builder->factory->session
- 代理对象，执行操作

新建Maven-webapp工程

构建项目

骨架构建：Create from archetype

- org.apache.maven.archetypes:maven-archetype-**webapp**



解决webapp项目构建慢的问题：Properties+

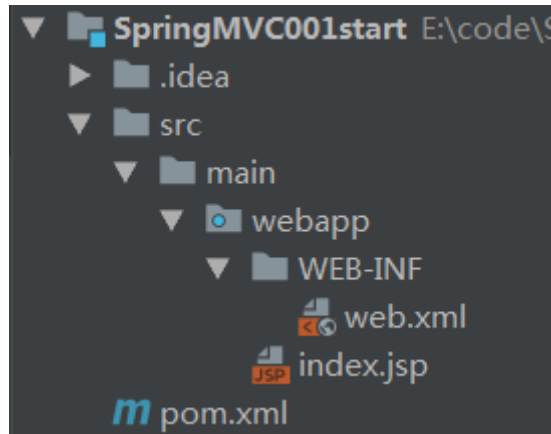
- archetypeCatalog
- internal

完善项目结构

构建完的项目，目录结构是不全的。

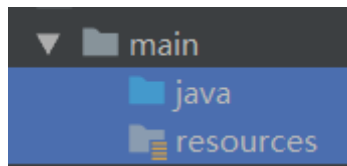
- src
 - main
 - webapp
 - WEB-INF

- web.xml
- index.jsp
- pom.xml

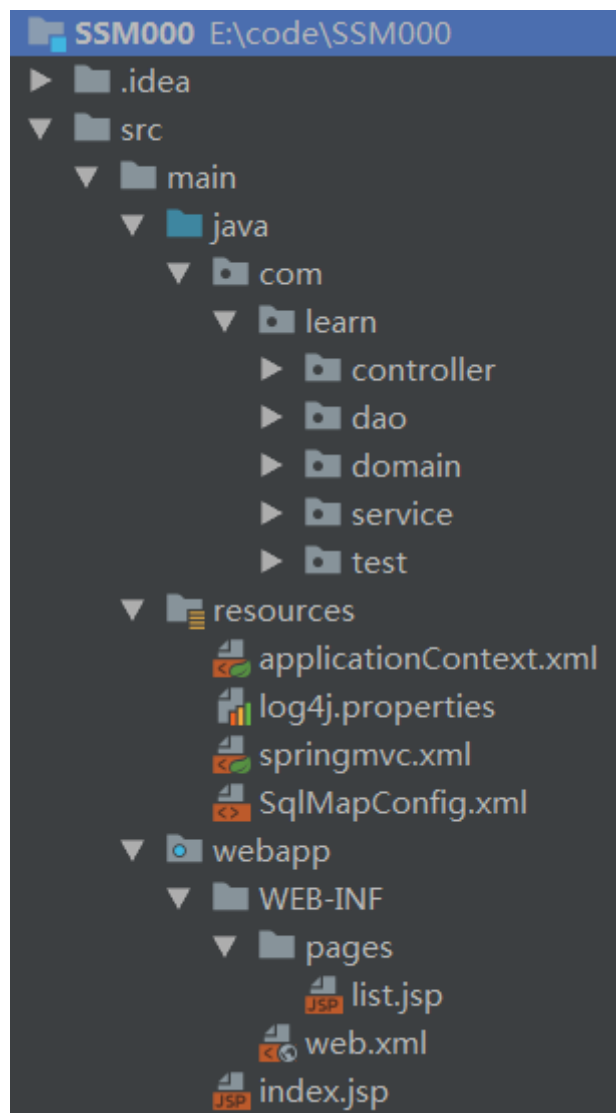


main下新建：

- java
 - 右键：Mark Directory as
 - Sources Root：源码根目录
- resources
 - 右键：Mark Directory as
 - Resources Root：资源根目录



最终结构



- src
 - main
 - java...com.learn
 - controller
 - dao
 - domain
 - service
 - test
 - resources
 - webapp
 - WEB-INF
 - index.jsp
- pom.xml

pom依赖

- aspectjweaver
- spring-aop
- spring-context

- spring-web
 - spring-webmvc
 - servlet-api
 - jsp-api
-
- jstl
-
- spring-test
-
- mysql-connector-java
 - c3p0
-
- spring-tx
 - spring-jdbc
-
- mybatis
 - mybatis-spring
-
- junit
-
- log4j
 - slf4j-api
 - slf4j-log4j12

```

1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <project xmlns="http://maven.apache.org/POM/4.0.0"
4   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
6     http://maven.apache.org/xsd/maven-4.0.0.xsd">
7   <modelVersion>4.0.0</modelVersion>
8
9   <groupId>com.grape</groupId>
10  <artifactId>SSM000</artifactId>
11  <version>1.0-SNAPSHOT</version>
12  <packaging>war</packaging>
13
14  <name>SSM000 Maven Webapp</name>
15  <!-- FIXME change it to the project's website -->
16  <url>http://www.example.com</url>
17
18  <properties>
19    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
20    <maven.compiler.source>1.7</maven.compiler.source>
21    <maven.compiler.target>1.7</maven.compiler.target>

```

```
20 <spring.version>5.0.2.RELEASE</spring.version>
21 <slf4j.version>1.6.6</slf4j.version>
22 <log4j.version>1.2.12</log4j.version>
23 <mysql.version>5.1.6</mysql.version>
24 <mybatis.version>3.4.5</mybatis.version>
25 </properties>
26
27
28 <dependencies>
29
30 <!-- spring -->
31 <dependency>
32 <groupId>org.aspectj</groupId>
33 <artifactId>aspectjweaver</artifactId>
34 <version>1.6.8</version>
35 </dependency>
36 <dependency>
37 <groupId>org.springframework</groupId>
38 <artifactId>spring-aop</artifactId>
39 <version>${spring.version}</version>
40 </dependency>
41
42 <dependency>
43 <groupId>org.springframework</groupId>
44 <artifactId>spring-context</artifactId>
45 <version>${spring.version}</version>
46 </dependency>
47
48 <dependency>
49 <groupId>org.springframework</groupId>
50 <artifactId>spring-web</artifactId>
51 <version>${spring.version}</version>
52 </dependency>
53 <dependency>
54 <groupId>org.springframework</groupId>
55 <artifactId>spring-webmvc</artifactId>
56 <version>${spring.version}</version>
57 </dependency>
58 <dependency>
59 <groupId>javax.servlet</groupId>
60 <artifactId>servlet-api</artifactId>
61 <version>2.5</version>
62 <scope>provided</scope>
63 </dependency>
64 <dependency>
65 <groupId>javax.servlet.jsp</groupId>
66 <artifactId>jsp-api</artifactId>
67 <version>2.0</version>
68 <scope>provided</scope>
69 </dependency>
70
71 <!-- 这个... -->
72 <dependency>
73 <groupId>jstl</groupId>
74 <artifactId>jstl</artifactId>
75 <version>1.2</version>
76 </dependency>
77
```

```
78 <dependency>
79   <groupId>org.springframework</groupId>
80   <artifactId>spring-test</artifactId>
81   <version>${spring.version}</version>
82 </dependency>
83
84 <dependency>
85   <groupId>org.springframework</groupId>
86   <artifactId>spring-tx</artifactId>
87   <version>${spring.version}</version>
88 </dependency>
89 <dependency>
90   <groupId>org.springframework</groupId>
91   <artifactId>spring-jdbc</artifactId>
92   <version>${spring.version}</version>
93 </dependency>
94 <dependency>
95   <groupId>mysql</groupId>
96   <artifactId>mysql-connector-java</artifactId>
97   <!-- 用5.几的就有问题，可能我MySQL的版本高，不兼容吧 -->
98   <version>8.0.15</version>
99 </dependency>
100
101 <dependency>
102   <groupId>junit</groupId>
103   <artifactId>junit</artifactId>
104   <version>4.12</version>
105   <scope>compile</scope>
106 </dependency>
107
108 <!-- log start -->
109 <dependency>
110   <groupId>log4j</groupId>
111   <artifactId>log4j</artifactId>
112   <version>${log4j.version}</version>
113 </dependency>
114 <dependency>
115   <groupId>org.slf4j</groupId>
116   <artifactId>slf4j-api</artifactId>
117   <version>${slf4j.version}</version>
118 </dependency>
119 <dependency>
120   <groupId>org.slf4j</groupId>
121   <artifactId>slf4j-log4j12</artifactId>
122   <version>${slf4j.version}</version>
123 </dependency>
124 <!-- log end -->
125
126 <dependency>
127   <groupId>org.mybatis</groupId>
128   <artifactId>mybatis</artifactId>
129   <version>${mybatis.version}</version>
130 </dependency>
131 <!-- 用于整合 -->
132 <dependency>
133   <groupId>org.mybatis</groupId>
134   <artifactId>mybatis-spring</artifactId>
135   <version>1.3.0</version>
```

```

136     </dependency>
137     <dependency>
138         <groupId>c3p0</groupId>
139         <artifactId>c3p0</artifactId>
140         <version>0.9.1.2</version>
141         <type>jar</type>
142         <scope>compile</scope>
143     </dependency>
144 </dependencies>
145
146 <build>
147     <finalName>SSM000</finalName>
148     <pluginManagement><!-- lock down plugins versions to avoid using Maven
defaults (may be moved to parent pom) -->
149     <plugins>
150         <plugin>
151             <artifactId>maven-clean-plugin</artifactId>
152             <version>3.1.0</version>
153         </plugin>
154         <!-- see http://maven.apache.org/ref/current/maven-core/default-
bindings.html#Plugin_bindings_for_war_packaging -->
155         <plugin>
156             <artifactId>maven-resources-plugin</artifactId>
157             <version>3.0.2</version>
158         </plugin>
159         <plugin>
160             <artifactId>maven-compiler-plugin</artifactId>
161             <version>3.8.0</version>
162         </plugin>
163         <plugin>
164             <artifactId>maven-surefire-plugin</artifactId>
165             <version>2.22.1</version>
166         </plugin>
167         <plugin>
168             <artifactId>maven-war-plugin</artifactId>
169             <version>3.2.2</version>
170         </plugin>
171         <plugin>
172             <artifactId>maven-install-plugin</artifactId>
173             <version>2.5.2</version>
174         </plugin>
175         <plugin>
176             <artifactId>maven-deploy-plugin</artifactId>
177             <version>2.8.2</version>
178         </plugin>
179     </plugins>
180 </pluginManagement>
181 </build>
182 </project>

```

独立Spring

SpringIOC容器管理：

- service
- dao

domain.Account

```
1  /**
2   * 账户
3   */
4  public class Account implements Serializable {
5
6      private Integer id;
7      private String name;
8      private Double money;
9
10     public Integer getId() {
11         return id;
12     }
13
14     public void setId(Integer id) {
15         this.id = id;
16     }
17
18     public String getName() {
19         return name;
20     }
21
22     public void setName(String name) {
23         this.name = name;
24     }
25
26     public Double getMoney() {
27         return money;
28     }
29
30     public void setMoney(Double money) {
31         this.money = money;
32     }
33
34     @Override
35     public String toString() {
36         return "Account{" +
37             "id=" + id +
38             ", name='" + name + '\'' +
39             ", money=" + money +
40             '}';
41     }
42 }
```

service.AccountService

```
1  public interface AccountService {
2
3      /**
4       * 查询所有账户
5       * @return
6       */
```

```

7     public List<Account> findAll();
8
9     /**
10    * 保存账户信息
11    * @param account
12    */
13    public void saveAccount(Account account);
14
15 }

```

service.impl.AccountServiceImpl

- 加注解@Service
- 注册，交给IOC容器进行管理

```

1  @Service("accountService")
2  public class AccountServiceImpl implements AccountService {
3
4      @Override
5      public List<Account> findAll() {
6          System.out.println("业务层: 查询所有账户");
7          return null;
8      }
9
10     @Override
11     public void saveAccount(Account account) {
12         System.out.println("业务层: 保存账户");
13     }
14 }

```

resources.applicationContext.xml

开启注解扫描，Spring只负责 业务层、持久层。

- service
- dao

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <beans xmlns="http://www.springframework.org/schema/beans"
3         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4         xmlns:context="http://www.springframework.org/schema/context"
5         xmlns:aop="http://www.springframework.org/schema/aop"
6         xmlns:tx="http://www.springframework.org/schema/tx"
7         xsi:schemaLocation="http://www.springframework.org/schema/beans
8
9         http://www.springframework.org/schema/beans/spring-beans.xsd
10                                http://www.springframework.org/schema/context
11                                http://www.springframework.org/schema/context/spring-context.xsd
12                                http://www.springframework.org/schema/aop
13                                http://www.springframework.org/schema/aop/spring-aop.xsd
14                                http://www.springframework.org/schema/tx

```

```

14 http://www.springframework.org/schema/tx/spring-tx.xsd"
15 >
16 <!-- 开启注解扫描，要扫描的是service和dao层的注解，要忽略web层注解，因为web层让
    SpringMVC框架去管理 -->
17 <context:component-scan base-package="com.learn">
18     <!-- 配置 哪些注解 不扫描 -->
19     <context:exclude-filter type="annotation"
    expression="org.springframework.stereotype.Controller"/>
20 </context:component-scan>
21 </beans>

```

test.TestSpring

```

1 public class TestSpring {
2
3     @Test
4     public void run1() {
5         //加载配置文件
6         ApplicationContext context = new
    ClassPathXmlApplicationContext("classpath:applicationContext.xml");
7         //获取对象
8         AccountService service = context.getBean("accountService",
    AccountService.class);
9         //调用方法
10        service.findAll();
11    }
12
13 }

```

独立SpringMVC

resources.springmvc.xml

- 开启注解扫描，SpringMVC负责web层
 - controller
- 配置 视图解析器
- 静态资源 (不过滤)
- 开启SpringMVC注解支持

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans xmlns="http://www.springframework.org/schema/beans"
3       xmlns:mvc="http://www.springframework.org/schema/mvc"

```

```

4      xmlns:context="http://www.springframework.org/schema/context"
5      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
6      xsi:schemaLocation="
7          http://www.springframework.org/schema/beans
8          http://www.springframework.org/schema/beans/spring-
beans.xsd
9          http://www.springframework.org/schema/mvc
10         http://www.springframework.org/schema/mvc/spring-
mvc.xsd
11         http://www.springframework.org/schema/context
12         http://www.springframework.org/schema/context/spring-
context.xsd">
13
14     <!-- 开启注解扫描,只扫描Controller注解 -->
15     <context:component-scan base-package="com.learn">
16         <context:include-filter type="annotation"
expression="org.springframework.stereotype.Controller"/>
17     </context:component-scan>
18
19     <!-- 配置视图解析器 -->
20     <bean id="internalResourceViewResolver"
class="org.springframework.web.servlet.view.InternalResourceViewResolver">
21         <property name="prefix" value="/WEB-INF/pages/"></property>
22         <property name="suffix" value=".jsp"></property>
23     </bean>
24
25
26     <!-- 设置静态资源 不过滤 -->
27     <mvc:resources location="/css/" mapping="/css/**" />
28     <mvc:resources location="/images/" mapping="/images/**" />
29     <mvc:resources location="/js/" mapping="/js/**" />
30
31     <!-- 开启SpringMVC注解的支持 -->
32     <mvc:annotation-driven></mvc:annotation-driven>
33 </beans>

```

webapp.WEB-INF.web.xml

- 前端控制器
 - 加载springmvc.xml配置文件
 - 启动服务器,创建servlet
- 编码过滤器

```

1 <!DOCTYPE web-app PUBLIC
2     "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
3     "http://java.sun.com/dtd/web-app_2_3.dtd" >
4
5 <web-app>
6     <display-name>Archetype Created Web Application</display-name>
7
8     <!-- 配置前端控制器 -->
9     <servlet>
10         <servlet-name>dispatcherServlet</servlet-name>
11         <servlet-
class>org.springframework.web.servlet.DispatcherServlet</servlet-class>

```



```

12      <!-- 加载SpringMVC.xml配置文件 -->
13      <init-param>
14          <param-name>contextConfigLocation</param-name>
15          <param-value>classpath:springmvc.xml</param-value>
16      </init-param>
17      <!-- 启动服务器 创建该Servlet对象 -->
18      <load-on-startup>1</load-on-startup>
19  </servlet>
20  <servlet-mapping>
21      <servlet-name>dispatcherServlet</servlet-name>
22      <url-pattern>/</url-pattern>
23  </servlet-mapping>
24
25  <!-- 解决中文乱码的过滤器 -->
26  <filter>
27      <filter-name>characterEncodingFilter</filter-name>
28      <filter-
class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>
29      <!-- 设置编码集 -->
30      <init-param>
31          <param-name>encoding</param-name>
32          <param-value>UTF-8</param-value>
33      </init-param>
34  </filter>
35  <filter-mapping>
36      <filter-name>characterEncodingFilter</filter-name>
37      <url-pattern>/*</url-pattern>
38  </filter-mapping>
39
40  </web-app>

```

webapp.index.jsp

```

1  <%@ page contentType="text/html; charset=UTF-8" language="java" %>
2  <html>
3  <head>
4      <title>Title</title>
5  </head>
6  <body>
7      <a href="account/findAll">测试SpringMVC 测试查询</a>
8  </body>
9  </html>

```

webapp.WEB-INF.pages.list.jsp

```

1 <%@ page contentType="text/html; charset=UTF-8" language="java"
  isELIgnored="false" %>
2 <%-- 引入 jstl的标签库 --%>
3 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
4 <html>
5 <head>
6     <title>Title</title>
7 </head>
8 <body>
9     <h3>查询了所有的账户信息</h3>
10 </body>
11 </html>

```

controller. AccountController

```

1 /**
2  * web层
3  */
4 @Controller
5 @RequestMapping("/account")
6 public class AccountController {
7
8     @RequestMapping("/findAll")
9     public String findAll() {
10         System.out.println("表现层: 查询所有账户信息");
11         return "list";//跳到list.jsp
12     }
13
14 }

```

独立MyBatis

dao. AccountDao

```

1 /**
2  * 账户dao接口
3  */
4 public interface AccountDao {
5
6     /**
7      * 查询所有账户
8      * @return
9      */
10     @Select("select * from account")
11     List<Account> findAll();
12
13     /**
14      * 保存账户信息
15      * @param account
16      */
17     @Insert("insert into account (name,money) values (#{name}, #{money})")
18     void saveAccount(Account account);

```

```
19  
20 }
```

resources.SqlMapConfig.xml

```
1 <?xml version="1.0" encoding="UTF-8"?>  
2 <!DOCTYPE configuration  
3     PUBLIC "-//mybatis.org//DTD Config 3.0//EN"  
4     "http://mybatis.org/dtd/mybatis-3-config.dtd">  
5 <configuration>  
6     <environments default="mysql">  
7         <environment id="mysql">  
8             <transactionManager type="JDBC"/>  
9             <dataSource type="POOLED">  
10                 <property name="driver" value="com.mysql.cj.jdbc.Driver"/>  
11                 <property name="url"  
value="jdbc:mysql://localhost:3306/db"/>  
12                 <property name="username" value="root"/>  
13                 <property name="password" value="xxxx"/>  
14             </dataSource>  
15         </environment>  
16     </environments>  
17  
18     <!-- 引入映射配置:xml | 注解 -->  
19     <mappers>  
20         <!-- <mapper class="com.learn.dao.AccountDao"/> -->  
21         <!-- 该包下所有的dao接口都可以使用 -->  
22         <package name="com.learn.dao"/>  
23     </mappers>  
24 </configuration>
```

test.MyBatis

```
1 public class TestMyBatis {  
2  
3     /**  
4     * 测试查询  
5     * @throws Exception  
6     */  
7     @Test  
8     public void testFindAll() throws Exception {  
9         //1.加载配置文件  
10         InputStream in = Resources.getResourceAsStream("SqlMapConfig.xml");  
11         //2.创建工厂  
12         SqlSessionFactory factory = new  
SqlSessionFactoryBuilder().build(in);  
13         //3.创建session  
14         SqlSession session = factory.openSession();  
15  
16         //4.获取代理对象  
17         AccountDao dao = session.getMapper(AccountDao.class);  
18         //5.查询  
19         List<Account> accounts = dao.findAll();
```

```

20         for (Account account : accounts) {
21             System.out.println(account);
22         }
23
24         //释放资源
25         session.close();
26         in.close();
27     }
28
29     @Test
30     public void testSaveAccount() throws Exception {
31         //1.加载配置文件
32         InputStream in = Resources.getResourceAsStream("SqlMapConfig.xml");
33         //2.创建工厂
34         SqlSessionFactory factory = new
35         SqlSessionFactoryBuilder().build(in);
36         //3.创建session
37         SqlSession session = factory.openSession();
38
39         //4.获取代理对象
40         AccountDao dao = session.getMapper(AccountDao.class);
41
42         Account account = new Account();
43         account.setName("史睿生");
44         account.setMoney(3000d);
45
46         dao.saveAccount(account);
47         session.commit();
48
49         //释放资源
50         session.close();
51         in.close();
52     }
53 }

```

Spring整合SpringMVC

- 在controller中能调用service(由IOC管理)代表整合成功
- ? : 启动tomcat, 根据 web.xml 做初始化
 - 前端控制器
 - 加载springmvc.xml
 - ? Spring.xml的配置文件没加载过

Controller

- controller里调用service(IOC管理), 整合成功
- 加入成员 accountService, 注入
- 调用service方法

```

1  /**
2  * web层

```

```

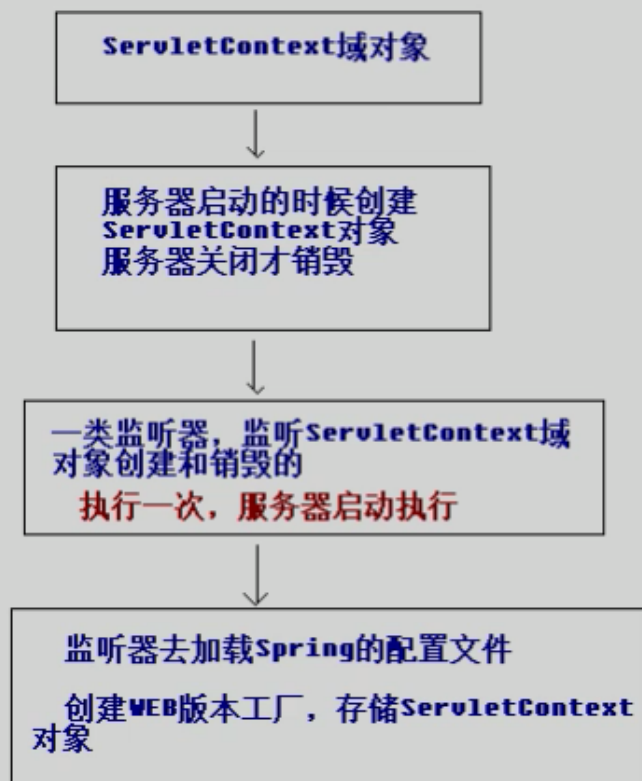
3  */
4  @Controller
5  @RequestMapping("/account")
6  public class AccountController {
7      @Autowired
8      private AccountService accountService;
9
10     @RequestMapping("/findAll")
11     public String findAll() {
12         System.out.println("表现层: 查询所有账户信息");
13         //调用service的方法
14         accountService.findAll();
15         return "list";//跳到list.jsp
16     }
17
18 }

```

web.xml

- 监听器
 - 依赖 spring-web
 - 这里用于 加载 Spring.xml 其实是 applicationContext.xml

Spring整合SpringMVC：启动Tomcat服务器的时候，需要加载spring的配置文件



```

1  <!DOCTYPE web-app PUBLIC
2  "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
3  "http://java.sun.com/dtd/web-app_2_3.dtd" >
4
5  <web-app>

```

```

6     <display-name>Archetype Created Web Application</display-name>
7
8     <!-- 监听器 加载Spring.xml -->
9     <!-- 默认只加载WEB-INF目录下的applicationContext.xml配置文件：1、复制一份过来
10    2、传参数 -->
11    <listener>
12        <listener-
13            class>org.springframework.web.context.ContextLoaderListener</listener-
14            class>
15        </listener>
16    <!-- 设置文件的路径 -->
17    <context-param>
18        <param-name>contextConfigLocation</param-name>
19        <param-value>classpath:applicationContext.xml</param-value>
20    </context-param>
21
22    <!-- 配置前端控制器 -->
23    <servlet>
24        <servlet-name>dispatcherServlet</servlet-name>
25        <servlet-
26            class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
27        <!-- 加载SpringMVC.xml配置文件 -->
28        <init-param>
29            <param-name>contextConfigLocation</param-name>
30            <param-value>classpath:springmvc.xml</param-value>
31        </init-param>
32        <!-- 启动服务器 创建该Servlet对象 -->
33        <load-on-startup>1</load-on-startup>
34    </servlet>
35    <servlet-mapping>
36        <servlet-name>dispatcherServlet</servlet-name>
37        <url-pattern>/</url-pattern>
38    </servlet-mapping>
39
40    <!-- 解决中文乱码的过滤器 -->
41    <filter>
42        <filter-name>characterEncodingFilter</filter-name>
43        <filter-
44            class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>
45        <!-- 设置编码集 -->
46        <init-param>
47            <param-name>encoding</param-name>
48            <param-value>UTF-8</param-value>
49        </init-param>
50    </filter>
51    <filter-mapping>
52        <filter-name>characterEncodingFilter</filter-name>
53        <url-pattern>/*</url-pattern>
54    </filter-mapping>
55
56 </web-app>

```

Spring整合MyBatis

Service能调用到dao对象。

dao 注册到 IOC中：

- 以前：测试类，自己加载SqlMapConfig.xml，生成代理
- 现在：代理对象 存到 IOC容器中

SqlMapConfig.xml废弃，内容转移到 applicationContext.xml

resources\applicationContext.xml

不用SqlMapConfig.xml了。

- 连接池
- 工厂对象SqlSessionFactory
 - 通过org.mybatis.spring.SqlSessionFactoryBean
- AccountDao接口所在包
 - 通过org.mybatis.spring.mapper.MapperScannerConfigurer
- 该有的都有了，别的就不需要我们操心了，Spring会自己做的

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans xmlns="http://www.springframework.org/schema/beans"
3       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4       xmlns:context="http://www.springframework.org/schema/context"
5       xmlns:aop="http://www.springframework.org/schema/aop"
6       xmlns:tx="http://www.springframework.org/schema/tx"
7       xsi:schemaLocation="http://www.springframework.org/schema/beans
8
9         http://www.springframework.org/schema/beans/spring-beans.xsd
10        http://www.springframework.org/schema/context
11
12        http://www.springframework.org/schema/context/spring-context.xsd
13        http://www.springframework.org/schema/aop
14
15        http://www.springframework.org/schema/aop/spring-aop.xsd
16        http://www.springframework.org/schema/tx
17
18        http://www.springframework.org/schema/tx/spring-tx.xsd"
19 >
20 <!-- 开启注解扫描，要扫描的是service和dao层的注解，要忽略web层注解，因为web层让
21 SpringMVC框架去管理 -->
22 <context:component-scan base-package="com.learn">
23   <!-- 配置要忽略的注解 -->
24   <context:exclude-filter type="annotation"
25     expression="org.springframework.stereotype.Controller"/>
26 </context:component-scan>
27
28 <!-- Spring整合MyBatis框架 SqlMapConfig的转移? -->
29 <!-- 配置连接池 C3P0 -->
30 <bean id="dataSource"
31     class="com.mchange.v2.c3p0.ComboPooledDataSource">
32   <property name="driverClass" value="com.mysql.cj.jdbc.Driver"/>
33   <property name="jdbcUrl" value="jdbc:mysql://localhost:3306/db"/>
34   <property name="user" value="root"/>
35   <property name="password" value="xxxx"/>
36 </bean>
```

```

29     </bean>
30
31     <!-- 配置SqlSessionFactory工厂对象 -->
32     <bean id="sqlSessionFactory"
33         class="org.mybatis.spring.SqlSessionFactoryBean">
34         <property name="dataSource" ref="dataSource"></property>
35     </bean>
36
37     <!-- 配置AccountDao接口所在的包 映射扫描配置的类 -->
38     <bean id="mapperScanner"
39         class="org.mybatis.spring.mapper.MapperScannerConfigurer">
40         <property name="basePackage" value="com.learn.dao"></property>
41     </bean>
42 </beans>

```

dao.AccountDao

- 加个注解，注册一下。@Repository

```

1  /**
2   * 账户dao接口
3   */
4  @Repository//整合配置后才有用
5  public interface AccountDao {
6
7      /**
8       * 查询所有账户
9       * @return
10      */
11      @Select("select * from account")
12      List<Account> findAll();
13
14      /**
15       * 保存账户信息
16       * @param account
17       */
18      @Insert("insert into account (name,money) values (#{name}, #{money})")
19      void saveAccount(Account account);
20
21  }

```

service.impl.AccountServiceImpl

- service 能调用 dao 说明 整合成功
- 加入成员 dao
- 调用dao方法

```

1  @Service("accountService")
2  public class AccountServiceImpl implements AccountService {
3
4      @Autowired
5      private AccountDao accountDao;

```



```

6
7     @Override
8     public List<Account> findAll() {
9         System.out.println("业务层: 查询所有账户");
10        return accountDao.findAll();
11    }
12
13    @Override
14    public void saveAccount(Account account) {
15        System.out.println("业务层: 保存账户");
16        accountDao.saveAccount(account);
17    }
18 }

```

测试

controller.AccountController

- 能查到数据了
- 把数据 展示到 页面上
- Model
- 存数据

```

1  /**
2   * web层
3   */
4  @Controller
5  @RequestMapping("/account")
6  public class AccountController {
7      @Autowired
8      private AccountService accountService;
9
10     @RequestMapping("/findAll")
11     public String findAll(Model model) {
12         System.out.println("表现层: 查询所有账户信息");
13         //调用service的方法
14         List<Account> accounts = accountService.findAll();
15         model.addAttribute("accounts", accounts);
16         return "list";//跳到list.jsp
17     }
18
19 }

```

webapp.WEB-INF.pages.list.jsp

- 展示 查询到的数据
- isELIgnored="false" 开启表达式
- 使用 jstl

```

1  <%@ page contentType="text/html; charset=UTF-8" language="java"
   isELIgnored="false" %>
2  <!-- 引入 jstl的标签库 -->
3  <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
4  <html>
5  <head>
6      <title>Title</title>
7  </head>
8  <body>
9      <h3>查询了所有的账户信息</h3>
10
11     <c:forEach items="${accounts}" var="account">
12         ${account.name}
13     </c:forEach>
14
15 </body>
16 </html>

```

加入Spring声明式事务管理

resources.applicationContext.xml

- 事务管理器
 - DataSourceTransactionManager
- 事务通知
 - 方法名 & 事务
- AOP

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <beans xmlns="http://www.springframework.org/schema/beans"
3      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4      xmlns:context="http://www.springframework.org/schema/context"
5      xmlns:aop="http://www.springframework.org/schema/aop"
6      xmlns:tx="http://www.springframework.org/schema/tx"
7      xsi:schemaLocation="http://www.springframework.org/schema/beans
8
9      http://www.springframework.org/schema/beans/spring-beans.xsd
10         http://www.springframework.org/schema/context
11
12         http://www.springframework.org/schema/context/spring-context.xsd
13         http://www.springframework.org/schema/aop
14
15         http://www.springframework.org/schema/aop/spring-aop.xsd
16         http://www.springframework.org/schema/tx
17
18         http://www.springframework.org/schema/tx/spring-tx.xsd"
19 >
20 <!-- 开启注解扫描，要扫描的是service和dao层的注解，要忽略web层注解，因为web层让
   SpringMVC框架去管理 -->
21     <context:component-scan base-package="com.learn">
22         <!-- 配置要忽略的注解 -->
23         <context:exclude-filter type="annotation"
24             expression="org.springframework.stereotype.Controller"/>
25     </context:component-scan>

```

```

21
22 <!-- Spring整合MyBatis框架 SqlMapConfig的转移? -->
23 <!-- 配置连接池 C3P0 -->
24 <bean id="dataSource"
25 class="com.mchange.v2.c3p0.ComboPooledDataSource">
26     <property name="driverClass" value="com.mysql.cj.jdbc.Driver"/>
27     <property name="jdbcUrl" value="jdbc:mysql://localhost:3306/db"/>
28     <property name="user" value="root"/>
29     <property name="password" value="xxxx"/>
30 </bean>
31
32 <!-- 配置SqlSessionFactory工厂对象 -->
33 <bean id="sqlSessionFactory"
34 class="org.mybatis.spring.SqlSessionFactoryBean">
35     <property name="dataSource" ref="dataSource"/></property>
36 </bean>
37
38 <!-- 配置AccountDao接口所在的包 映射扫描配置的类 -->
39 <bean id="mapperScanner"
40 class="org.mybatis.spring.mapper.MapperScannerConfigurer">
41     <property name="basePackage" value="com.learn.dao"/></property>
42 </bean>
43
44 <!-- 配置Spring声明式事务管理 -->
45 <!-- 配置事务管理器 -->
46 <bean id="transactionManager"
47 class="org.springframework.jdbc.datasource.DataSourceTransactionManager">
48     <property name="dataSource" ref="dataSource"/></property>
49 </bean>
50
51 <!-- 配置事务通知 -->
52 <tx:advice id="txAdvice" transaction-manager="transactionManager">
53     <tx:attributes>
54         <!-- find开头的方法 只读事务 -->
55         <tx:method name="find*" read-only="true"/>
56
57         <tx:method name="*" isolation="DEFAULT"/></tx:method>
58     </tx:attributes>
59 </tx:advice>
60
61 <!-- 配置AOP增强 -->
62 <aop:config>
63     <aop:advisor advice-ref="txAdvice" pointcut="execution(*
64 com.learn.service.impl.*ServiceImpl.*(..))"/></aop:advisor>
65 </aop:config>
66 </beans>

```

webapp. index.jsp

试试保存

```

1 <%@ page contentType="text/html; charset=UTF-8" language="java" %>
2 <html>
3 <head>

```

```

4      <title>Title</title>
5  </head>
6  <body>
7      <a href="account/findAll">测试SpringMVC 测试查询</a>
8
9      <h3>测试保存</h3>
10     <form action="account/saveAccount" method="post">
11         姓名:<input type="text" name="name"><br>
12         金额:<input type="text" name="money"><br>
13         <input type="submit" value="保存">
14     </form>
15
16 </body>
17 </html>

```

controller. AccountController

- save方法，接收 表单为 对象
- 保存
- 重定向 到 list.jsp

```

1  /**
2   * web层
3   */
4  @Controller
5  @RequestMapping("/account")
6  public class AccountController {
7
8      /**
9       * 这里能调用 service 就算 springmvc 和 spring整合成功了
10      * 这里启动方式是 tomcat, web.xml只配置了加载springmvc.xml, 别的不扫描
11
12      * 加载springConfig.xml
13      */
14
15      @Autowired
16      private AccountService accountService;
17
18
19      @RequestMapping("/findAll")
20      public String findAll(Model model) {
21          System.out.println("表现层: 查询所有账户信息");
22          List<Account> accounts = accountService.findAll();
23          model.addAttribute("accounts", accounts); //存到域对象里
24          return "list"; //跳到list.jsp
25      }
26
27      @RequestMapping("/saveAccount")
28      public void saveAccount(Account account, HttpServletRequest request,
29      HttpServletResponse response) throws IOException {
30          System.out.println("表现层: 保存账户");
31          accountService.saveAccount(account);
32
33          response.sendRedirect(request.getContextPath()+"/account/findAll"); //保存完
34          就去查

```

```
32         return;
33     }
34
35 }
```

java.com.learn

controller

AccountController

1SpringMVC

```
1  /**
2   * web层
3   */
4  @Controller
5  @RequestMapping("/account")
6  public class AccountController {
7
8      @RequestMapping("/findAll")
9      public String findAll() {
10         System.out.println("表现层: 查询所有账户信息");
11         return "list";//跳到list.jsp
12     }
13
14 }
```

2Spring整合SpringMVC

- controller里调用service(IOC管理), 整合成功
- 加入成员 accountService, 注入
- 调用service方法

```
1  /**
2   * web层
3   */
4  @Controller
5  @RequestMapping("/account")
6  public class AccountController {
7      @Autowired
8      private AccountService accountService;
9
10     @RequestMapping("/findAll")
11     public String findAll() {
12         System.out.println("表现层: 查询所有账户信息");
13         //调用service的方法
14         accountService.findAll();
15         return "list";//跳到list.jsp
16     }
17
18 }
```

3Spring整合MyBatis

- 能查到数据了
- 把数据 展示到 页面上

```
1  /**
2   * web层
3   */
4  @Controller
5  @RequestMapping("/account")
6  public class AccountController {
7      @Autowired
8      private AccountService accountService;
9
10     @RequestMapping("/findAll")
11     public String findAll(Model model) {
12         System.out.println("表现层: 查询所有账户信息");
13         //调用service的方法
14         List<Account> accounts = accountService.findAll();
15         model.addAttribute("accounts", accounts);
16         return "list";//跳到list.jsp
17     }
18
19
20
21 }
```

4声明式事务

- save方法, 接收 表单为 对象
- 保存
- 重定向 到 list.jsp

```
1  /**
2   * web层
3   */
4  @Controller
5  @RequestMapping("/account")
6  public class AccountController {
7
8      /**
9       * 这里能调用 service 就算 springmvc 和 spring整合成功了
10      * 这里启动方式是 tomcat, web.xml只配置了加载springmvc.xml, 别的不扫描
11
12      * 加载springConfig.xml
13      */
14
15     @Autowired
16     private AccountService accountService;
17
18
19     @RequestMapping("/findAll")
20     public String findAll(Model model) {
21         System.out.println("表现层: 查询所有账户信息");
22         List<Account> accounts = accountService.findAll();
23         model.addAttribute("accounts", accounts);//存到域对象里
24         return "list";//跳到list.jsp
25     }
26 }
```

```

25     }
26
27     @RequestMapping("/saveAccount")
28     public void saveAccount(Account account, HttpServletRequest request,
29     HttpServletResponse response) throws IOException {
30         System.out.println("表现层: 保存账户");
31         accountService.saveAccount(account);
32
33         response.sendRedirect(request.getContextPath()+"/account/findAll");//保存完
34         就去查
35     }
36     }
37 }

```

dao

AccountDao.java

1MyBatis

```

1  /**
2   * 账户dao接口
3   */
4  public interface AccountDao {
5
6      /**
7       * 查询所有账户
8       * @return
9       */
10     @Select("select * from account")
11     List<Account> findAll();
12
13     /**
14      * 保存账户信息
15      * @param account
16      */
17     @Insert("insert into account (name,money) values (#{name}, #{money})")
18     void saveAccount(Account account);
19
20 }

```

2Spring整合MyBatis

- 加个注解注册一下@Repository

```

1  /**
2   * 账户dao接口
3   */
4  @Repository//整合配置后才有用
5  public interface AccountDao {
6
7      /**
8       * 查询所有账户
9       * @return

```

```

10     */
11     @Select("select * from account")
12     List<Account> findAll();
13
14     /**
15      * 保存账户信息
16      * @param account
17      */
18     @Insert("insert into account (name,money) values (#{name}, #{money})")
19     void saveAccount(Account account);
20
21 }

```

domain

Account.java

```

1  /**
2   * 账户
3   */
4  public class Account implements Serializable {
5
6      private Integer id;
7      private String name;
8      private Double money;
9
10     public Integer getId() {
11         return id;
12     }
13
14     public void setId(Integer id) {
15         this.id = id;
16     }
17
18     public String getName() {
19         return name;
20     }
21
22     public void setName(String name) {
23         this.name = name;
24     }
25
26     public Double getMoney() {
27         return money;
28     }
29
30     public void setMoney(Double money) {
31         this.money = money;
32     }
33
34     @Override
35     public String toString() {
36         return "Account{" +
37             "id=" + id +
38             ", name='" + name + '\'' +

```



```

39         ", money=" + money +
40         "}'";
41     }
42 }

```

service

AccountService.java

```

1  public interface AccountService {
2
3      /**
4       * 查询所有账户
5       * @return
6       */
7      public List<Account> findAll();
8
9      /**
10     * 保存账户信息
11     * @param account
12     */
13     public void saveAccount(Account account);
14
15 }

```

impl

AccountServiceImpl.java

```

1  public class AccountServiceImpl implements AccountService {
2
3      @Override
4      public List<Account> findAll() {
5          System.out.println("业务层: 查询所有账户");
6          return null;
7      }
8
9      @Override
10     public void saveAccount(Account account) {
11         System.out.println("业务层: 保存账户");
12     }
13 }

```

1Spring

- 加注解@Service
- 注册，交给IOC容器进行管理

```

1  @Service("accountService")
2  public class AccountServiceImpl implements AccountService {
3
4      @Override

```

```

5     public List<Account> findAll() {
6         System.out.println("业务层: 查询所有账户");
7         return null;
8     }
9
10    @Override
11    public void saveAccount(Account account) {
12        System.out.println("业务层: 保存账户");
13    }
14 }

```

2Spring整合MyBatis

- service 能 调用 dao 说明 整合成功
- 加入 成员 dao
- 调用dao方法

```

1  @Service("accountService")
2  public class AccountServiceImpl implements AccountService {
3
4      @Autowired
5      private AccountDao accountDao;
6
7      @Override
8      public List<Account> findAll() {
9          System.out.println("业务层: 查询所有账户");
10         return accountDao.findAll();
11     }
12
13     @Override
14     public void saveAccount(Account account) {
15         System.out.println("业务层: 保存账户");
16         accountDao.saveAccount(account);
17     }
18 }

```

test

TestSpring

```

1 public class TestSpring {
2
3     @Test
4     public void run1() {
5         //加载配置文件
6         ApplicationContext context = new
ClassPathXmlApplicationContext("classpath:applicationContext.xml");
7         //获取对象
8         AccountService service = context.getBean("accountService",
AccountService.class);
9         //调用方法
10        service.findAll();
11    }
12
13 }

```

TestMyBatis

```

1 public class TestMyBatis {
2
3     /**
4      * 测试查询
5      * @throws Exception
6      */
7     @Test
8     public void testFindAll() throws Exception {
9         //1.加载配置文件
10        InputStream in = Resources.getResourceAsStream("SqlMapConfig.xml");
11        //2.创建工厂
12        SqlSessionFactory factory = new
SqlSessionFactoryBuilder().build(in);
13        //3.创建session
14        SqlSession session = factory.openSession();
15
16        //4.获取代理对象
17        AccountDao dao = session.getMapper(AccountDao.class);
18        //5.查询
19        List<Account> accounts = dao.findAll();
20        for (Account account : accounts) {
21            System.out.println(account);
22        }
23
24        //释放资源
25        session.close();
26        in.close();
27    }
28
29    @Test
30    public void testSaveAccount() throws Exception {
31        //1.加载配置文件
32        InputStream in = Resources.getResourceAsStream("SqlMapConfig.xml");
33        //2.创建工厂
34        SqlSessionFactory factory = new
SqlSessionFactoryBuilder().build(in);
35        //3.创建session

```

```

36         SqlSession session = factory.openSession();
37
38         //4. 获取代理对象
39         AccountDao dao = session.getMapper(AccountDao.class);
40
41         Account account = new Account();
42         account.setName("史睿生");
43         account.setMoney(3000d);
44
45         dao.saveAccount(account);
46         session.commit();
47
48         //释放资源
49         session.close();
50         in.close();
51
52     }
53
54 }

```

resources

applicationContext.xml

1.Spring

- 开启注解扫描，Spring只负责 业务层、持久层。
 - service
 - dao

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <beans xmlns="http://www.springframework.org/schema/beans"
3         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4         xmlns:context="http://www.springframework.org/schema/context"
5         xmlns:aop="http://www.springframework.org/schema/aop"
6         xmlns:tx="http://www.springframework.org/schema/tx"
7         xsi:schemaLocation="http://www.springframework.org/schema/beans
8
9         http://www.springframework.org/schema/beans/spring-beans.xsd
10         http://www.springframework.org/schema/context
11
12         http://www.springframework.org/schema/context/spring-context.xsd
13         http://www.springframework.org/schema/aop
14
15         http://www.springframework.org/schema/aop/spring-aop.xsd
16         http://www.springframework.org/schema/tx
17
18         http://www.springframework.org/schema/tx/spring-tx.xsd"
19  >
20  <!-- 开启注解扫描，要扫描的是service和dao层的注解，要忽略web层注解，因为web层让
21  SpringMVC框架去管理 -->
22  <context:component-scan base-package="com.learn">
23      <!-- 配置 哪些注解 不扫描 -->

```

```

19     <context:exclude-filter type="annotation"
expression="org.springframework.stereotype.Controller"/>
20     </context:component-scan>
21 </beans>

```

2Spring整合MyBatis

- 连接池
- 工厂对象SqlSessionFactory
- AccountDao接口所在包

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans xmlns="http://www.springframework.org/schema/beans"
3     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4     xmlns:context="http://www.springframework.org/schema/context"
5     xmlns:aop="http://www.springframework.org/schema/aop"
6     xmlns:tx="http://www.springframework.org/schema/tx"
7     xsi:schemaLocation="http://www.springframework.org/schema/beans
8         http://www.springframework.org/schema/beans/spring-beans.xsd
9         http://www.springframework.org/schema/context
10            http://www.springframework.org/schema/context/spring-context.xsd
11            http://www.springframework.org/schema/aop
12            http://www.springframework.org/schema/aop/spring-aop.xsd
13            http://www.springframework.org/schema/tx
14            http://www.springframework.org/schema/tx/spring-tx.xsd"
15 >
16 <!-- 开启注解扫描，要扫描的是service和dao层的注解，要忽略web层注解，因为web层让
SpringMVC框架去管理 -->
17     <context:component-scan base-package="com.learn">
18         <!-- 配置要忽略的注解 -->
19         <context:exclude-filter type="annotation"
expression="org.springframework.stereotype.Controller"/>
20     </context:component-scan>
21
22 <!-- Spring整合MyBatis框架 SqlMapConfig的转移? -->
23 <!-- 配置连接池 C3P0 -->
24 <bean id="dataSource"
class="com.mchange.v2.c3p0.ComboPooledDataSource">
25     <property name="driverClass" value="com.mysql.cj.jdbc.Driver"/>
26     <property name="jdbcUrl" value="jdbc:mysql://localhost:3306/db"/>
27     <property name="user" value="root"/>
28     <property name="password" value="xxxx"/>
29 </bean>
30
31 <!-- 配置SqlSessionFactory工厂对象 -->
32 <bean id="sqlSessionFactory"
class="org.mybatis.spring.SqlSessionFactoryBean">
33     <property name="dataSource" ref="dataSource"/></property>
34 </bean>
35
36 <!-- 配置AccountDao接口所在的包 映射扫描配置类 -->
37 <bean id="mapperScanner"
class="org.mybatis.spring.mapper.MapperScannerConfigurer">

```

```

38         <property name="basePackage" value="com.learn.dao"></property>
39     </bean>
40
41 </beans>

```

3Spring声明式事务管理

- 事务管理器
- 事务通知
- AOP

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <beans xmlns="http://www.springframework.org/schema/beans"
3      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4      xmlns:context="http://www.springframework.org/schema/context"
5      xmlns:aop="http://www.springframework.org/schema/aop"
6      xmlns:tx="http://www.springframework.org/schema/tx"
7      xsi:schemaLocation="http://www.springframework.org/schema/beans
8
9          http://www.springframework.org/schema/beans/spring-beans.xsd
10             http://www.springframework.org/schema/context
11
12             http://www.springframework.org/schema/context/spring-context.xsd
13             http://www.springframework.org/schema/aop
14
15             http://www.springframework.org/schema/aop/spring-aop.xsd
16             http://www.springframework.org/schema/tx
17
18             http://www.springframework.org/schema/tx/spring-tx.xsd"
19  >
20  <!-- 开启注解扫描，要扫描的是service和dao层的注解，要忽略web层注解，因为web层让
21  SpringMVC框架去管理 -->
22      <context:component-scan base-package="com.learn">
23          <!-- 配置要忽略的注解 -->
24          <context:exclude-filter type="annotation"
25              expression="org.springframework.stereotype.Controller"/>
26      </context:component-scan>
27
28  <!-- Spring整合MyBatis框架 SqlMapConfig的转移? -->
29      <!-- 配置连接池 C3P0 -->
30      <bean id="dataSource"
31          class="com.mchange.v2.c3p0.ComboPooledDataSource">
32          <property name="driverClass" value="com.mysql.cj.jdbc.Driver"/>
33          <property name="jdbcUrl" value="jdbc:mysql://localhost:3306/db"/>
34          <property name="user" value="root"/>
35          <property name="password" value="xxxx"/>
36      </bean>
37
38      <!-- 配置SqlSessionFactory工厂对象 -->
39      <bean id="sqlSessionFactory"
40          class="org.mybatis.spring.SqlSessionFactoryBean">
41          <property name="dataSource" ref="dataSource"></property>
42      </bean>
43
44      <!-- 配置AccountDao接口所在的包 映射扫描配置类 -->
45      <bean id="mapperScanner"
46          class="org.mybatis.spring.mapper.MapperScannerConfigurer">

```

```

38         <property name="basePackage" value="com.learn.dao"></property>
39     </bean>
40
41     <!-- 配置Spring声明式事务管理 -->
42     <!-- 配置事务管理器 -->
43     <bean id="transactionManager"
44 class="org.springframework.jdbc.datasource.DataSourceTransactionManager">
45         <property name="dataSource" ref="dataSource"></property>
46     </bean>
47
48     <!-- 配置事务通知 -->
49     <tx:advice id="txAdvice" transaction-manager="transactionManager">
50         <tx:attributes>
51             <!-- find开头的方法 只读事务 -->
52             <tx:method name="find*" read-only="true"/>
53
54             <tx:method name="*" isolation="DEFAULT"></tx:method>
55         </tx:attributes>
56     </tx:advice>
57
58     <!-- 配置AOP增强 -->
59     <aop:config>
60         <aop:advisor advice-ref="txAdvice" pointcut="execution(*
61 com.learn.service.impl.*ServiceImpl.*(..))"></aop:advisor>
62     </aop:config>
63
64 </beans>

```

springmvc.xml

1SpringMVC

- 开启注解扫描，SpringMVC负责web层
 - controller
- 配置 视图解析器
- 静态资源 (不过滤)
- 开启SpringMVC注解支持

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans xmlns="http://www.springframework.org/schema/beans"
3       xmlns:mvc="http://www.springframework.org/schema/mvc"
4       xmlns:context="http://www.springframework.org/schema/context"
5       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
6       xsi:schemaLocation="
7           http://www.springframework.org/schema/beans
8           http://www.springframework.org/schema/beans/spring-
9 beans.xsd
10           http://www.springframework.org/schema/mvc
11           http://www.springframework.org/schema/mvc/spring-
12 mvc.xsd
13           http://www.springframework.org/schema/context
14           http://www.springframework.org/schema/context/spring-
15 context.xsd">

```

```

14      <!-- 开启注解扫描,只扫描Controller注解 -->
15      <context:component-scan base-package="com.learn">
16          <context:include-filter type="annotation"
17      expression="org.springframework.stereotype.Controller"/>
18      </context:component-scan>
19
20      <!-- 配置视图解析器 -->
21      <bean id="internalResourceViewResolver"
22      class="org.springframework.web.servlet.view.InternalResourceViewResolver">
23          <property name="prefix" value="/WEB-INF/pages/"></property>
24          <property name="suffix" value=".jsp"></property>
25      </bean>
26
27      <!-- 设置静态资源 不过滤 -->
28      <mvc:resources location="/css/" mapping="/css/**" />
29      <mvc:resources location="/images/" mapping="/images/**" />
30      <mvc:resources location="/js/" mapping="/js/**" />
31
32      <!-- 开启SpringMVC注解的支持 -->
33      <mvc:annotation-driven></mvc:annotation-driven>
34  </beans>

```

SqlMapConfig.xml

1MyBatis

- 数据源|环境
- 映射 位置 指定

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <!DOCTYPE configuration
3      PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
4      "http://mybatis.org/dtd/mybatis-3-config.dtd">
5  <configuration>
6      <environments default="mysql">
7          <environment id="mysql">
8              <transactionManager type="JDBC"/>
9              <dataSource type="POOLED">
10                 <property name="driver" value="com.mysql.cj.jdbc.Driver"/>
11                 <property name="url"
12     value="jdbc:mysql://localhost:3306/db"/>
13                 <property name="username" value="root"/>
14                 <property name="password" value="xxxx"/>
15             </dataSource>
16         </environment>
17     </environments>
18
19     <!-- 引入映射配置:xml|注解 -->
20     <mappers>
21         <!-- <mapper class="com.learn.dao.AccountDao"/> -->
22         <!-- 该包下所有的dao接口都可以使用 -->
23         <package name="com.learn.dao"/>
24     </mappers>
25 </configuration>

```


log4j.properties

```
1 # Set root category priority to INFO and its only appender to CONSOLE.
2 #log4j.rootCategory=INFO, CONSOLE          debug   info   warn error
   fatal
3 log4j.rootCategory=info, CONSOLE, LOGFILE
4
5 # Set the enterprise logger category to FATAL and its only appender to
   CONSOLE.
6 log4j.logger.org.apache.axis.enterprise=FATAL, CONSOLE
7
8 # CONSOLE is set to be a ConsoleAppender using a PatternLayout.
9 log4j.appender.CONSOLE=org.apache.log4j.ConsoleAppender
10 log4j.appender.CONSOLE.layout=org.apache.log4j.PatternLayout
11 log4j.appender.CONSOLE.layout.ConversionPattern=%d{ISO8601} %-6r [%15.15t]
   %-5p %30.30c %x - %m\n
12
13 # LOGFILE is set to be a File appender using a PatternLayout.
14 log4j.appender.LOGFILE=org.apache.log4j.FileAppender
15 log4j.appender.LOGFILE.File=d:\axis.log
16 log4j.appender.LOGFILE.Append=true
17 log4j.appender.LOGFILE.layout=org.apache.log4j.PatternLayout
18 log4j.appender.LOGFILE.layout.ConversionPattern=%d{ISO8601} %-6r [%15.15t]
   %-5p %30.30c %x - %m\n
19
20
```

webapp

index.jsp

1SpringMVC

做测试用的，Spring有测试类，这个需要jsp

```
1 <%@ page contentType="text/html; charset=UTF-8" language="java" %>
2 <html>
3 <head>
4     <title>Title</title>
5 </head>
6 <body>
7     <a href="account/findAll">测试SpringMVC 测试查询</a>
8 </body>
9 </html>
```

2声明式事务

```
1 <%@ page contentType="text/html; charset=UTF-8" language="java" %>
2 <html>
3 <head>
```

```

4      <title>Title</title>
5  </head>
6  <body>
7      <a href="account/findAll">测试SpringMVC 测试查询</a>
8
9      <h3>测试保存</h3>
10     <form action="account/saveAccount" method="post">
11         姓名:<input type="text" name="name"><br>
12         金额:<input type="text" name="money"><br>
13         <input type="submit" value="保存">
14     </form>
15
16 </body>
17 </html>

```

WEB-INF

web.xml

1SpringMVC

- 前端控制器
 - 加载springmvc.xml配置文件
 - 启动服务器，创建servlet
- 编码过滤器

```

1  <!DOCTYPE web-app PUBLIC
2      "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
3      "http://java.sun.com/dtd/web-app_2_3.dtd" >
4
5  <web-app>
6      <display-name>Archetype Created web Application</display-name>
7
8
9      <!-- 配置前端控制器 -->
10     <servlet>
11         <servlet-name>dispatcherServlet</servlet-name>
12         <servlet-
13 class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
14         <!-- 加载SpringMVC.xml配置文件 -->
15         <init-param>
16             <param-name>contextConfigLocation</param-name>
17             <param-value>classpath:springmvc.xml</param-value>
18         </init-param>
19         <!-- 启动服务器 创建该Servlet对象 -->
20         <load-on-startup>1</load-on-startup>
21     </servlet>
22     <servlet-mapping>
23         <servlet-name>dispatcherServlet</servlet-name>
24         <url-pattern>/</url-pattern>
25     </servlet-mapping>
26
27     <!-- 解决中文乱码的过滤器 -->
28     <filter>
29         <filter-name>characterEncodingFilter</filter-name>

```

```

29     <filter-
class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>
30     <!-- 设置编码集 -->
31     <init-param>
32         <param-name>encoding</param-name>
33         <param-value>UTF-8</param-value>
34     </init-param>
35 </filter>
36 <filter-mapping>
37     <filter-name>characterEncodingFilter</filter-name>
38     <url-pattern>/*</url-pattern>
39 </filter-mapping>
40
41 </web-app>

```

2Spring整合SpringMVC

- 配置 监听器 ContextLoaderListener
 - 加载Spring.xml
 - 依赖 pom中 spring-web
- 问题：Listener 默认只加载 WEB-INF下的文件
 - 复制一份 到WEB-INF下
 - 传参数

```

1  <!DOCTYPE web-app PUBLIC
2  "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
3  "http://java.sun.com/dtd/web-app_2_3.dtd" >
4
5  <web-app>
6      <display-name>Archetype Created Web Application</display-name>
7
8      <!-- 监听器 加载Spring.xml -->
9      <!-- 默认只加载WEB-INF目录下的applicationContext.xml配置文件：1、复制一份过来
10      2、传参数 -->
11      <listener>
12          <listener-
class>org.springframework.web.context.ContextLoaderListener</listener-
class>
13      </listener>
14      <!-- 设置配置文件的路径 -->
15      <context-param>
16          <param-name>contextConfigLocation</param-name>
17          <param-value>classpath:applicationContext.xml</param-value>
18      </context-param>
19
20      <!-- 配置前端控制器 -->
21      <servlet>
22          <servlet-name>dispatcherServlet</servlet-name>
23          <servlet-
class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
24      <!-- 加载SpringMVC.xml配置文件 -->
25      <init-param>
26          <param-name>contextConfigLocation</param-name>
27          <param-value>classpath:springmvc.xml</param-value>
28      </init-param>

```

```

29      <!-- 启动服务器 创建该Servlet对象 -->
30      <load-on-startup>1</load-on-startup>
31  </servlet>
32  <servlet-mapping>
33      <servlet-name>dispatcherServlet</servlet-name>
34      <url-pattern>/</url-pattern>
35  </servlet-mapping>
36
37  <!-- 解决中文乱码的过滤器 -->
38  <filter>
39      <filter-name>characterEncodingFilter</filter-name>
40      <filter-
41  <class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>
42      <!-- 设置编码集 -->
43      <init-param>
44          <param-name>encoding</param-name>
45          <param-value>UTF-8</param-value>
46      </init-param>
47  </filter>
48  <filter-mapping>
49      <filter-name>characterEncodingFilter</filter-name>
50      <url-pattern>/*</url-pattern>
51  </filter-mapping>
52 </web-app>

```

pages

list.jsp

1SpringMVC

```

1  <%@ page contentType="text/html; charset=UTF-8" language="java" %>
2  <%-- 引入 jstl的标签库 --%>
3  <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
4  <html>
5  <head>
6      <title>Title</title>
7  </head>
8  <body>
9      <h3>查询了所有的账户信息</h3>
10 </body>
11 </html>

```

2Spring整合MyBatis

- 展示 查询到的数据
- isELIgnored="false" 开启表达式
- 使用 jstl

```

1  <%@ page contentType="text/html; charset=UTF-8" language="java"
    isELIgnored="false" %>
2  <%-- 引入 jstl的标签库 --%>
3  <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

```

```
4 <html>
5 <head>
6   <title>Title</title>
7 </head>
8 <body>
9   <h3>查询了所有的账户信息</h3>
10
11   <c:forEach items="${accounts}" var="account">
12     ${account.name}
13   </c:forEach>
14
15 </body>
16 </html>
```

其他知识

- resources目录 等于 java工程的 src目录下，类路径下
- 会到 WEB-INF/classes/..

EOF
