

百知教育 — Spring系列课程 — MVC框架整合

第一章、MVC框架整合思想

1. 搭建Web运行环境

```
1  <dependency>
2    <groupId>javax.servlet</groupId>
3    <artifactId>javax.servlet-api</artifactId>
4    <version>3.1.0</version>
5    <scope>provided</scope>
6  </dependency>
7
8  <dependency>
9    <groupId>javax.servlet</groupId>
10   <artifactId>jstl</artifactId>
11   <version>1.2</version>
12 </dependency>
13
14 <!--
15   https://mvnrepository.com/artifact/javax.servlet.jsp/javax.servlet.jsp-api -->
16 <dependency>
17   <groupId>javax.servlet.jsp</groupId>
18   <artifactId>javax.servlet.jsp-api</artifactId>
19   <version>2.3.1</version>
20   <scope>provided</scope>
21 </dependency>
22
23 <dependency>
24   <groupId>org.springframework</groupId>
25   <artifactId>spring-web</artifactId>
26   <version>5.1.14.RELEASE</version>
27 </dependency>
28
29 <dependency>
30   <groupId>org.springframework</groupId>
31   <artifactId>spring-core</artifactId>
32   <version>5.1.14.RELEASE</version>
33 </dependency>
34
35 <dependency>
36   <groupId>org.springframework</groupId>
37   <artifactId>spring-beans</artifactId>
38   <version>5.1.14.RELEASE</version>
39 </dependency>
40
41 <dependency>
42   <groupId>org.springframework</groupId>
```

```
43     <artifactId>spring-tx</artifactId>
44     <version>5.1.14.RELEASE</version>
45 </dependency>
46
47 <dependency>
48     <groupId>org.springframework</groupId>
49     <artifactId>spring-jdbc</artifactId>
50     <version>5.1.14.RELEASE</version>
51 </dependency>
52
53 <dependency>
54     <groupId>org.mybatis</groupId>
55     <artifactId>mybatis-spring</artifactId>
56     <version>2.0.2</version>
57 </dependency>
58
59 <dependency>
60     <groupId>com.alibaba</groupId>
61     <artifactId>druid</artifactId>
62     <version>1.1.18</version>
63 </dependency>
64
65 <dependency>
66     <groupId>mysql</groupId>
67     <artifactId>mysql-connector-java</artifactId>
68     <version>5.1.48</version>
69 </dependency>
70
71 <dependency>
72     <groupId>org.mybatis</groupId>
73     <artifactId>mybatis</artifactId>
74     <version>3.4.6</version>
75 </dependency>
76
77 <dependency>
78     <groupId>junit</groupId>
79     <artifactId>junit</artifactId>
80     <version>4.11</version>
81     <scope>test</scope>
82 </dependency>
83
84 <!-- https://mvnrepository.com/artifact/org.springframework/spring-
context -->
85 <dependency>
86     <groupId>org.springframework</groupId>
87     <artifactId>spring-context</artifactId>
88     <version>5.1.4.RELEASE</version>
89 </dependency>
90
91 <dependency>
92     <groupId>org.springframework</groupId>
93     <artifactId>spring-aop</artifactId>
94     <version>5.1.14.RELEASE</version>
95 </dependency>
96
```

```

97     <dependency>
98         <groupId>org.aspectj</groupId>
99         <artifactId>aspectjrt</artifactId>
100        <version>1.8.8</version>
101    </dependency>
102
103    <dependency>
104        <groupId>org.aspectj</groupId>
105        <artifactId>aspectjweaver</artifactId>
106        <version>1.8.3</version>
107    </dependency>
108
109    <dependency>
110        <groupId>org.slf4j</groupId>
111        <artifactId>slf4j-log4j12</artifactId>
112        <version>1.7.25</version>
113    </dependency>
114
115    <dependency>
116        <groupId>log4j</groupId>
117        <artifactId>log4j</artifactId>
118        <version>1.2.17</version>
119    </dependency>

```

2. 为什么要整合MVC框架

- 1 1. MVC框架提供了控制器(Controller)调用Service
DAO ---》 Service
- 2 2. 请求响应的处理
- 3 3. 接受请求参数 request.getParameter("")
- 4 4. 控制程序的运行流程
- 5 5. 视图解析 (JSP JSON Freemarker Thymeleaf)

3. Spring可以整合那些MVC框架

- 1 1. struts1
- 2 2. webwork
- 3 3. jsf
- 4 4. struts2
- 5 5. springMVC

4.Spring整合MVC框架的核心思路

1. 准备工厂

- 1 1. Web开发过程中如何创建工厂
- 2 ApplicationContext ctx = new
ClassPathXmlApplicationContext("/applicationContext.xml");
- 3 WebXmlApplicationContext()
- 4 2. 如何保证工厂唯一同时被共用
- 5 被共用: Web request|session|ServletContext(application)
- 6 工厂存储在ServletContext这个作用域中
ServletContext.setAttribute("xxxx", ctx);
- 7

```

8      唯一: ServletContext对象 创建的同时 ---》 ApplicationContext ctx = new
      ClassPathXmlApplicationContext("/applicationContext.xml");
9
10     ServletContextListener ---> ApplicationContext ctx = new
      ClassPathXmlApplicationContext("/applicationContext.xml");
11     ServletContextListener 在ServletContext对象创建的同时, 被调用(只会
      被调用一次), 把工厂创建的代码, 写在ServletContextListener中, 也会保证只调用
12     一次, 最终工厂就保证了唯一性
13
14     3. 总结
15     ServletContextListener(唯一)
16     ApplicationContext ctx = new
      ClassPathXmlApplicationContext("/applicationContext.xml");
17     ServletContext.setAttribute("xxx", ctx) (共用)
18
19     4. Spring封装了一个ContextLoaderListener
20     1. 创建工厂
21     2. 把工厂存在ServletContext中

```

```

1 ContextLoaderListener使用方式
2
3 web.xml
4
5 <listener>
6     <listener-
      class>org.springframework.web.context.ContextLoaderListener</listener-
      class>
7 </listener>
8
9 <context-param>
10     <param-name>contextConfigLocation</param-name>
11     <param-value>classpath:applicationContext.xml</param-value>
12 </context-param>
13

```

2. 代码整合

1 依赖注入: 把Service对象注入个控制器对象。

<p>控制</p> <p>SpringMVC Controller Struts2 Action</p> <pre> public class XXXController XXXAction{ 1. 接受client请求参数 2. 调用Service对象 3. 流程跳转 (响应JSON) } </pre>	<pre> public class XXXController XXXAction{ private XXXService xxxService set get } </pre> <p>通过Spring进行依赖注入 配置文件中赋值</p>
---	---

第二章、Spring与Struts2框架整合 (选学)

1. Spring与Struts2整合思路分析

1 1. Struts2中的Action需要通过Spring的依赖注入获得Service对象。

2. Spring与Struts2整合的编码实现

- 搭建开发环境

- 引入相关jar (Spring Struts2)

```
1 <dependency>
2   <groupId>org.apache.struts</groupId>
3   <artifactId>struts2-spring-plugin</artifactId>
4   <version>2.3.8</version>
5 </dependency>
```

- 引入对应的配置文件

- applicationContext.xml
 - struts.xml
 - log4j.properties

- 初始化配置

```
1 <listener>
2   <listener-
3     class>org.springframework.web.context.ContextLoaderListener</l
4     istener-class>
5   </listener>
6
7   <context-param>
8     <param-name>contextConfigLocation</param-name>
9     <param-value>classpath:applicationContext.xml</param-value>
10  </context-param>
11
12  <filter>
13    <filter-name>struts2</filter-name>
14    <filter-
15      class>org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAnd
16      ExecuteFilter</filter-class>
17    </filter>
18    <filter-mapping>
19      <filter-name>struts2</filter-name>
20      <url-pattern>/*</url-pattern>
21    </filter-mapping>
```

- Spring (ContextLoaderListener → Web.xml)
 - Struts2(Filter → Web.xml)

- 编码

- 开发Service对象

```
1 最终在Spring配置文件中创建Service对象
2 <bean id="userService"
3     class="com.baizhi.struts2.UserServiceImp1"/>
```

- 开发Action对象

- 开发类

```

1  public class RegAction implements Action{
2      private UserService userService;
3      set get
4
5      public String execute(){
6          userService.register();
7          return Action.SUCCESS;
8      }
9  }

```

- Spring (applicationContext.xml)

```

1  <bean id="regAction" class="com.baizhi.struts2.RegAction"
    scope="prototype">
2      <property name="userService" ref="" />
3  </bean>

```

- Struts2(struts.xml)

```

1  <package name="ssm" extends="struts-default">
2      url reg.action ---> 会接受到用户的请求后，创建RegAction这个类
                           的对象 进行相应的处理
3
4      <action name="reg" class="regAction">
5          <result name="success">/index.jsp</result>
6      </action>
7  </package>

```

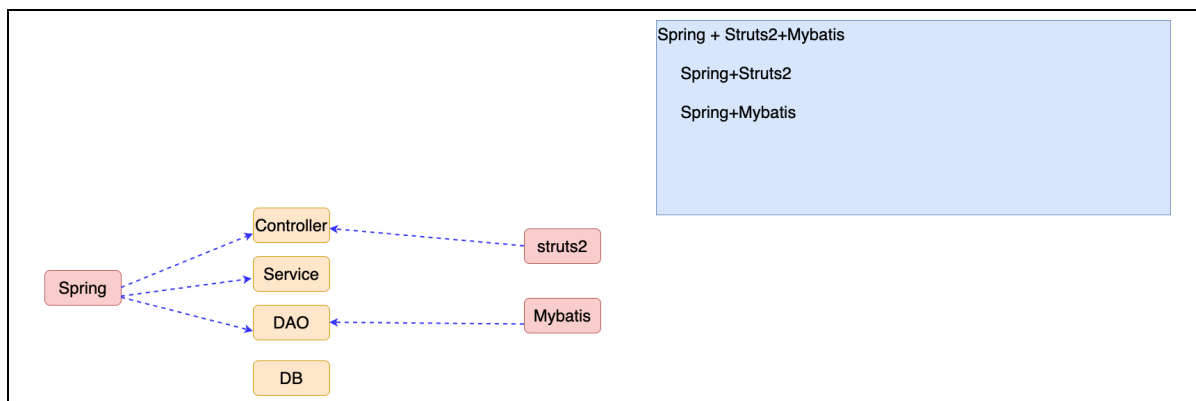
3. Spring+Struts2+Mybatis整合(SSM)

1. 思路分析

```

1  SSM = Spring+Struts2  Spring+Mybatis

```



2. 整合编码

- 搭建开发环境
 - 引入相关jar (Spring Struts2 Mybatis)

```

1
2

```

```
3      <dependency>
4          <groupId>org.apache.struts</groupId>
5          <artifactId>struts2-spring-plugin</artifactId>
6          <version>2.3.8</version>
7      </dependency>
8
9      <dependency>
10         <groupId>javax.servlet</groupId>
11         <artifactId>javax.servlet-api</artifactId>
12         <version>3.1.0</version>
13         <scope>provided</scope>
14     </dependency>
15
16     <dependency>
17         <groupId>javax.servlet</groupId>
18         <artifactId>jstl</artifactId>
19         <version>1.2</version>
20     </dependency>
21
22     <!--
23     https://mvnrepository.com/artifact/javax.servlet.jsp/javax.se
24     rvlet.jsp-api -->
25     <dependency>
26         <groupId>javax.servlet.jsp</groupId>
27         <artifactId>javax.servlet.jsp-api</artifactId>
28         <version>2.3.1</version>
29         <scope>provided</scope>
30     </dependency>
31
32     <dependency>
33         <groupId>org.springframework</groupId>
34         <artifactId>spring-web</artifactId>
35         <version>5.1.14.RELEASE</version>
36     </dependency>
37
38     <dependency>
39         <groupId>org.springframework</groupId>
40         <artifactId>spring-core</artifactId>
41         <version>5.1.14.RELEASE</version>
42     </dependency>
43
44     <dependency>
45         <groupId>org.springframework</groupId>
46         <artifactId>spring-beans</artifactId>
47         <version>5.1.14.RELEASE</version>
48     </dependency>
49
50     <dependency>
51         <groupId>org.springframework</groupId>
52         <artifactId>spring-tx</artifactId>
53         <version>5.1.14.RELEASE</version>
54     </dependency>
55
56     <dependency>
```

```

56     <groupId>org.springframework</groupId>
57     <artifactId>spring-jdbc</artifactId>
58     <version>5.1.14.RELEASE</version>
59 </dependency>
60
61 <dependency>
62     <groupId>org.mybatis</groupId>
63     <artifactId>mybatis-spring</artifactId>
64     <version>2.0.2</version>
65 </dependency>
66
67 <dependency>
68     <groupId>com.alibaba</groupId>
69     <artifactId>druid</artifactId>
70     <version>1.1.18</version>
71 </dependency>
72
73 <dependency>
74     <groupId>mysql</groupId>
75     <artifactId>mysql-connector-java</artifactId>
76     <version>5.1.48</version>
77 </dependency>
78
79 <dependency>
80     <groupId>org.mybatis</groupId>
81     <artifactId>mybatis</artifactId>
82     <version>3.4.6</version>
83 </dependency>
84
85 <dependency>
86     <groupId>junit</groupId>
87     <artifactId>junit</artifactId>
88     <version>4.11</version>
89     <scope>test</scope>
90 </dependency>
91
92 <!--
https://mvnrepository.com/artifact/org.springframework/spring
-context -->
93 <dependency>
94     <groupId>org.springframework</groupId>
95     <artifactId>spring-context</artifactId>
96     <version>5.1.4.RELEASE</version>
97 </dependency>
98
99 <dependency>
100     <groupId>org.springframework</groupId>
101     <artifactId>spring-aop</artifactId>
102     <version>5.1.14.RELEASE</version>
103 </dependency>
104
105 <dependency>
106     <groupId>org.aspectj</groupId>
107     <artifactId>aspectjrt</artifactId>
108     <version>1.8.8</version>

```



```

109     </dependency>
110
111     <dependency>
112         <groupId>org.aspectj</groupId>
113         <artifactId>aspectjweaver</artifactId>
114         <version>1.8.3</version>
115     </dependency>
116
117     <dependency>
118         <groupId>org.slf4j</groupId>
119         <artifactId>slf4j-log4j12</artifactId>
120         <version>1.7.25</version>
121     </dependency>
122
123     <dependency>
124         <groupId>log4j</groupId>
125         <artifactId>log4j</artifactId>
126         <version>1.2.17</version>
127     </dependency>
128
129
130
131

```

- 引入对应的配置文件

- applicationContext.xml
- struts.xml
- log4j.properties
- xxxxMapper.xml

- 初始化配置

- Spring (ContextLoaderListener → Web.xml)
- Struts2(Filter → Web.xml)

```

1  <listener>
2      <listener-
3      class>org.springframework.web.context.ContextLoaderListener</l
4      istener-class>
5  </listener>
6
7  <context-param>
8      <param-name>contextConfigLocation</param-name>
9      <param-value>classpath:applicationContext.xml</param-value>
10 </context-param>
11
12 <filter>
13     <filter-name>struts2</filter-name>
14     <filter-
15     class>org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAnd
16     ExecuteFilter</filter-class>
17 </filter>
18 <filter-mapping>
19     <filter-name>struts2</filter-name>
20     <url-pattern>/*</url-pattern>
21 </filter-mapping>

```

- 编码

- DAO (Spring+Mybatis)

```
1    1. 配置文件的配置
2        1. DataSource
3        2. SqlSessionFactory ----> SqlSessionFactoryBean
4            1. dataSource
5            2. typeAliasesPackage
6            3. mapperLocations
7        3. MapperScannerConfigurer ----> DAO接口实现类
8    2. 编码
9        1. entity
10       2. table
11       3. DAO接口
12       4. 实现Mapper文件
```

- Service (Spring添加事务)

```
1    1. 原始对象 ---》 注入DAO
2    2. 额外功能 ---》 DataSourceTransactionManager ----> dataSource
3    3. 切入点 + 事务属性
4        @Transactional(propagation,readOnly...)
5    4. 组装切面
6        <tx:annotation-driven
```

- Controller (Spring+Struts2)

```
1    1. 开发控制器 implements Action 注入Service
2    2. Spring的配置文件
3        1. 注入 Service
4        2. scope = prototype
5    3. struts.xml
6        <action class="spring配置文件中action对应的id值"/>
```

4. Spring开发过程中多配置文件的处理

```
1    Spring会根据需要，把配置信息分门别类的放置在多个配置文件中，便于后续的管理及维护。
2
3    DAO ----- applicationContext-dao.xml
4    Service ---  applicationContext-service.xml
5    Action ---   applicationContext-action.xml
6
7    注意：虽然提供了多个配置文件，但是后续应用的过程中，还要进行整合
```

- 通配符方式

```
1    1. 非web环境
2        ApplicationContext ctx = new
3        ClassPathXmlApplicationContext("/applicationContext-*.xml");
4    2. web环境
5        <context-param>
6            <param-name>contextConfigLocation</param-name>
7            <param-value>classpath:applicationContext-*.xml</param-
8            value>
9        </context-param>
```

- <import>标签

```
1  applicationContext.xml 目的 整合其他配置内容
2      <import resource="applicationContext-dao.xml " />
3      <import resource="applicationContext-service.xml " />
4      <import resource="applicationContext-action.xml " />
5
6  1. 非web环境
7      ApplicationContext ctx = new
ClassPathXmlApplicationContext("/applicationContext.xml");
8  2. web环境
9      <context-param>
10         <param-name>contextConfigLocation</param-name>
11         <param-value>classpath:applicationContext.xml</param-value>
12     </context-param>
13
```