# 08-SSM框架整合

1、三大框架整合

在老期的项目中，一般都是使用ssm项目做开发的，虽然现在的主流开发是springboot来做开发，但是ssm的基本整合还是需要掌握的。

### 整合SSM框架要做哪些事情：

- SpringMVC: pom
    1. web.xml
        a. 前端调度器servlet
        b. 编码过滤器filter
        c. 支持rest的过滤器
    2. springmvc.xml
        a. 扫描controller包
        b. 添加<annotation-driver>
        c. 视图解析器
        d. 静态资源解析
    3. 添加控制器类...
- Spring:
    1. web.xml
        a. 监听器（在启动web容器时加载）
    2. spring.xml
        a. 扫描所有除了controller包的其他包
        b. 声明式事
- MyBatis
    1. 需要和spring整合
        a. 将sqlSessionFactory 配置为spring的bean
            i. 数据源 配置为spring的bean
            ii. 全局配置文件
            iii. 所有mapper映射文件
        b. 将mapper接口的包交给spring
    2. 加入全局配置文件

## 1、导入pom文件

```
<?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/xs
```

```xml
    <modelVersion>4.0.0</modelVersion>

    <groupId>cn.tulingxueyuan</groupId>
    <artifactId>ssm</artifactId>
    <version>1.0-SNAPSHOT</version>

    <properties>
        <SPRING.VERSION>5.2.6.RELEASE</SPRING.VERSION>
        <ASPECTJWEAVER.VERSION>1.9.5</ASPECTJWEAVER.VERSION>
    </properties>

    <dependencies>
        <!--springmvc-->
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-webmvc</artifactId>
            <version>${SPRING.VERSION}</version>
        </dependency>
        <!--spring-->
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-aop</artifactId>
            <version>${SPRING.VERSION}</version>
        </dependency>
        <dependency>
            <groupId>org.aspectj</groupId>
            <artifactId>aspectjweaver</artifactId>
            <version>${ASPECTJWEAVER.VERSION}</version>
        </dependency>
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-aspects</artifactId>
            <version>${SPRING.VERSION}</version>
        </dependency>
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-orm</artifactId>
            <version>${SPRING.VERSION}</version>
        </dependency>

        <!--spring-mybatis 适配器-->
        <dependency>
            <groupId>org.mybatis</groupId>
            <artifactId>mybatis-spring</artifactId>
            <version>2.0.5</version>
        </dependency>


        <!--mybatis核心依赖-->
        <dependency>
            <groupId>org.mybatis</groupId>
            <artifactId>mybatis</artifactId>
            <version>3.5.5</version>
        </dependency>
        <!--连接池-->
        <dependency>
            <groupId>com.alibaba</groupId>
            <artifactId>druid</artifactId>
            <version>1.1.21</version>
        </dependency>
        <!--mysql 对应版本的连接器驱动-->
        <dependency>
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-java</artifactId>
            <version>5.1.47</version>
        </dependency>
        <!-- log start -->
        <dependency>
            <groupId>org.slf4j</groupId>
            <artifactId>slf4j-api</artifactId>
            <version>1.7.30</version>
        </dependency>


        <dependency>
            <groupId>ch.qos.logback</groupId>
            <artifactId>logback-classic</artifactId>
            <version>1.2.3</version>
        </dependency>
        <!-- log end -->


        <!-- https://mvnrepository.com/artifact/com.fasterxml.jackson.core/jackson-core
        <dependency>
            <groupId>com.fasterxml.jackson.core</groupId>
            <artifactId>jackson-core</artifactId>
            <version>2.10.3</version>
        </dependency>
        <!-- https://mvnrepository.com/artifact/com.fasterxml.jackson.core/jackson-datab
        <dependency>
            <groupId>com.fasterxml.jackson.core</groupId>
            <artifactId>jackson-databind</artifactId>
            <version>2.10.3</version>
        </dependency>
        <!-- https://mvnrepository.com/artifact/com.fasterxml.jackson.core/jackson-annot
        <dependency>
            <groupId>com.fasterxml.jackson.core</groupId>
            <artifactId>jackson-annotations</artifactId>
            <version>2.10.3</version>
        </dependency>

        <dependency>
            <groupId>junit</groupId>
            <artifactId>junit</artifactId>
            <version>4.12</version>
            <scope>test</scope>
        </dependency>
    </dependencies>

</project>
```

## 2、编写各个框架的配置文件

web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/
         version="3.1">
```

```


    <!--spring 基于web应用的启动-->
    <listener>
        <listener-class>org.springframework.web.context.ContextLoaderListener</listener-
    </listener>
    <!--全局参数：spring配置文件-->
    <context-param>
        <param-name>contextConfigLocation</param-name>
        <param-value>classpath:spring-core.xml</param-value>
    </context-param>



    <!--前端调度器servlet-->
    <servlet>
        <servlet-name>dispatcherServlet</servlet-name>
        <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
        <!--设置配置文件的路径-->
        <init-param>
            <param-name>contextConfigLocation</param-name>
            <param-value>classpath:spring-mvc.xml</param-value>
        </init-param>
        <!--设置启动即加载-->
        <load-on-startup>1</load-on-startup>
    </servlet>
    <servlet-mapping>
        <servlet-name>dispatcherServlet</servlet-name>
        <url-pattern>/</url-pattern>
    </servlet-mapping>

    <!--编码过滤器filter-->
    <filter>
        <filter-name>characterEncoding</filter-name>
        <filter-class>org.springframework.web.filter.CharacterEncodingFilter</filter-cla
        <!--设置编码格式-->
        <init-param>
            <param-name>encoding</param-name>
            <param-value>UTF-8</param-value>
        </init-param>
        <!---->
        <init-param>
            <param-name>forceEncoding</param-name>
            <param-value>true</param-value>
        </init-param>
    </filter>
    <filter-mapping>
        <filter-name>characterEncoding</filter-name>
        <servlet-name>dispatcherServlet</servlet-name>
    </filter-mapping>

    <!--支持rest的过滤器-->
    <filter>
        <filter-name>hiddenHttpMethod</filter-name>
        <filter-class>org.springframework.web.filter.HiddenHttpMethodFilter</filter-clas
    </filter>
    <filter-mapping>
        <filter-name>hiddenHttpMethod</filter-name>
        <servlet-name>dispatcherServlet</servlet-name>
    </filter-mapping>

</web-app>
```

springmvc.xml

```
<?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:context="http://www.springframework.org/schema/context"
       xmlns:mvc="http://www.springframework.org/schema/mvc"
       xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.spring

    <!--springmvc只扫描控制器-->
    <context:component-scan base-package="cn.tulingxueyuan" use-default-filters="false">
        <context:include-filter type="annotation" expression="org.springframework.stereo
    </context:component-scan>
    <!--静态资源的扫描-->
    <mvc:default-servlet-handler></mvc:default-servlet-handler>
    <!--动态资源的扫描-->
    <mvc:annotation-driven></mvc:annotation-driven>
    <!--配置视图管理器-->
    <bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">
        <property name="prefix" value="/WEB-INF/page/"></property>
        <property name="suffix" value=".jsp"></property>
    </bean>
</beans>
```

spring-core.xml

```
<?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:context="http://www.springframework.org/schema/context" xmlns:tx="http://www
       xmlns:aop="http://www.springframework.org/schema/aop" xmlns:mybatis="http://mybat
       xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.spring

    <!--扫描所有除了controller包的其他包-->
    <context:component-scan base-package="cn.tulingxueyuan">
        <context:exclude-filter type="annotation" expression="org.springframework.stereo
    </context:component-scan>


    <!--引入外部属性资源文件-->
    <context:property-placeholder location="classpath:db.properties"></context:property-

    <!--创建Druid数据源-->
    <bean class="com.alibaba.druid.pool.DruidDataSource" id="dataSource">
        <property name="username" value="${mysql.username}"></property>
        <property name="password" value="${mysql.password}"></property>
        <property name="url"  value="${mysql.url}"></property>
        <property name="driverClassName" value="${mysql.driverClassName}"></property>
    </bean>

    <!--声明式事-->
    <bean class="org.springframework.jdbc.datasource.DataSourceTransactionManager" id="t
        <property name="dataSource" ref="dataSource"></property>
    </bean>

    <!--基于驱动-->
    <tx:annotation-driven transaction-manager="transactionManager"></tx:annotation-drive


    <!--用于声明事务切入的所有方法-->
    <aop:config>
        <aop:pointcut id="transactionCut" expression="execution(* cn.tulingxueyuan.servi
    </aop:config>

    <!--用来明确切点匹配到的方法哪些方法需要使用事务-->
    <tx:advice id="advice" transaction-manager="transactionManager">
        <tx:attributes>
            <!--可以使用通配符-->
            <tx:method name="add*"/>
            <tx:method name="update*"/>
            <tx:method name="delete*"/>
            <tx:method name="get*" read-only="true" propagation="SUPPORTS"></tx:method>
            <tx:method name="query*" read-only="true"></tx:method>
        </tx:attributes>
    </tx:advice>

    <!--SqlSessionFactory-->
    <bean class="org.mybatis.spring.SqlSessionFactoryBean" id="sqlSessionFactory">
        <!-- 指定spring中的数据源 -->
        <property name="dataSource" ref="dataSource"></property>
        <property name="configLocation" value="classpath:mybatis-config.xml"></property>
        <property name="mapperLocations" value="classpath:cn/tulingxueyuan/mapper/*.xml"
    </bean>

    <!--将mapper接口交给spring管理-->
    <mybatis:scan base-package="cn.tulingxueyuan.mapper"></mybatis:scan>

</beans>
```

mybatis-config.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE configuration
```

```
        PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
```

```xml
        "http://mybatis.org/dtd/mybatis-3-config.dtd">
<!--就是DOCTYPE后面对应的根节点-->
<configuration>


    <!--mybatis的设置选项  可以改变mybatis运行时行为-->
    <settings>
        <setting name="mapUnderscoreToCamelCase" value="true"/>
    </settings>

    <!--类型别名可为 Java 类型设置一个缩写名字。 它仅用于 XML 配置，意在降低冗余的全限定类名书
    <typeAliases>
        <package name="cn.tulingxueyuan.pojo"/>
    </typeAliases>

</configuration>
```

db.properties

```
mysql.username = root
mysql.password=123456
mysql.url=jdbc:mysql://localhost:3306/mybatis
mysql.driverClassName=com.mysql.jdbc.Driver
```

生成 pojo\mapper接口\mapper映射文件

TestController.java

```java

/***
 * @Author 徐庶   QQ:1092002729
 * @Slogan 致敬大师，致敬未来的你
 */
@RestController
public class EmpController {


    @Autowired
    IEmpService empService;

    @RequestMapping("test")
    public  List<Emp> test(){
        List<Emp> emps = empService.selectEmp();
        return emps;
    }
}
```