1、为什么要整合?

1、Spring作为一个大容器，来管理所有的组件，然后利用Spring的IOC来解决组件之间的动态依赖与动态注入。（AutoWire）

避免每次进行CRUD之前还需要调用 sqlsessionFactory等等

2、同时还需要用spring来控制事务。

2、SSM项目目录结构：

以后mybatis-config可以写在spring文件中

3、配置步骤：

- **导包**

```
spring运行依赖的日志包
<artifactId>commons-logging</artifactId>
AOP功能（事务控制）
<artifactId>aopalliance</artifactId>
<artifactId>aspectjweaver</artifactId>
<artifactId>cglib</artifactId>
<artifactId>spring-aspects</artifactId>
<artifactId>spring-aop</artifactId>
ioc核心jar包
 <artifactId>spring-beans</artifactId>
 <artifactId>spring-core</artifactId>
 <artifactId>spring-expression</artifactId>
 <artifactId>spring-context</artifactId>
 jdbc数据库（tx-事务）
 <artifactId>spring-jdbc</artifactId>
 <artifactId>spring-orm</artifactId>
 <artifactId>spring-tx</artifactId>
 web模块-springMVC
 <artifactId>spring-web</artifactId>
 <artifactId>spring-webmvc</artifactId>
 mybatis与数据库驱动
 <artifactId>mybatis</artifactId>
 <artifactId>mysql-connector-java</artifactId>
 mybatis与spring整合的适配包
 <artifactId>mybatis-spring</artifactId>
 数据库连接池
 <artifactId>c3p0</artifactId>
 <artifactId>mchange-commons-java</artifactId>
 <artifactId>commons-dbcp</artifactId>
 JSTL
 <artifactId>taglibs-standard-impl</artifactId>
 <artifactId>taglibs-standard-spec</artifactId>
 工具
 <artifactId>lombok</artifactId>
 <artifactId>junit</artifactId>
 其他
 <artifactId>commons-pool</artifactId>
 <artifactId>ehcache-core</artifactId>
 <artifactId>mybatis-ehcache</artifactId>
 <artifactId>javax.servlet-api</artifactId>
 <artifactId>slf4j-api</artifactId>
 <artifactId>slf4j-log4j12</artifactId>
```

- **配置文件：**

  web配置：

  web.xml

```
<!--Spring配置-->
<!--启动ioc容器 （跟随web一起启动）-->
 <context-param>
 <param-name>contextConfigLocation</param-name>
 <param-value>classpath:applicationContext.xml</param-value>
 </context-param>
 <!--spring容器加载监听器-->
 <listener>
  <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
  </listener>

<!-- SpringMvc配置-->
<!-- 前端控制器-->
  <servlet>
  <servlet-name>spring</servlet-name>
  <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
  <load-on-startup>1</load-on-startup>
  </servlet>
<!-- 拦截所有请求-->
  <servlet-mapping>
  <servlet-name>spring</servlet-name>
  <url-pattern>/</url-pattern>
  </servlet-mapping>
```

spring-servlet.xml（只是控制网站跳转逻辑）

```
<!--扫描所有组件：-->
  <context:component-scan base-package="flipped">
  <!--只扫描控制器 include-filter/exclude-filter-->
  <context:include-filter type="annotation"
expression="org.springframework.stereotype.Controller"/>
  </context:component-scan>

<!--开挂-->
  <!-- 配置一个视图解析器：视图解析器，可以简化方法的返回值，返回值就是作为目标页面地址，帮我们拼接页面地址-->
  <!--类似mybatis拼接sql-->
   <bean class="org.springframework.web.servlet.view.InternalResourceViewResolver" id="InternalResourceViewResolver">
   <!--前缀-->
   <property name="prefix" value="/WEB-INF/pages/"/>
   <!--后缀-->
   <property name="suffix" value=".jsp"/>
   </bean>
   <mvc:annotation-driven/>
   <mvc:default-servlet-handler/>
```

mybatis.xml常用的设置项目：

```
<!-- 驼峰命名-->
  <settings>
  <setting name="mapUnderscoreToCamelCase" value="false"/>
```

```
</settings>
```

spring配置：（重点）

```
<!-- Spring希望管理所有的业务逻辑组件，等。。。-->
<context:component-scan base-package="flipped">
<!-- 除了控制器不需要-->
<context:exclude-filter type="annotation"
expression="org.springframework.stereotype.Controller"/>
</context:component-scan>

<!-- JdbcTemplate 对象 -->
<bean id="jdbcTemplate" class="org.springframework.jdbc.core.JdbcTemplate">
<!--注入 dataSource-->
<property name="dataSource" ref="dataSource"></property><!--set方式注入-->
</bean>

<!-- Spring用来控制业务逻辑。数据源、事务控制、aop-->
<!-- dataSource 配置 -->
<context:property-placeholder location="classpath:dbconfig.properties"/>

<bean id="dataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource">
<!-- 基本属性 url、user、password -->
<property name="jdbcUrl" value="${url}"/>
<property name="user" value="${jdbc.username}"/>
<property name="password" value="${password}"/>
<property name="driverClass" value="${driver}"></property>
</bean>


<!-- 整合mybatis-->
<bean id="dataSourceTransactionManager" class="org.springframework.jdbc.datasource.DataSourceTra
nsactionManager">
<!-- 指定数据库连接池中的连接-->
<property name="dataSource" ref="dataSource" ></property>
</bean>

<!-- 启用对事务注解的支持 -->
<tx:annotation-driven transaction-manager="dataSourceTransactionManager"/>


<!-- 配置Spring创建 sqlsessionFactory对象 -->
<bean id="sqlSessionFactoryBean" class="org.mybatis.spring.SqlSessionFactoryBean" >
<property name="dataSource" ref="dataSource"></property>
<!--configLocation指定mybatis全局配置文件的位置 可以在里面配置一些设置项-->
<property name="configLocation" value="classpath:mybatis-config.xml"></property>
<!--mapperLocations: 指定mapper文件的位置-->
<property name="mapperLocations" value="classpath:mapper/*.xml"></property>
</bean>

<!-- 扫描所有的mapper接口的实现，让这些mapper能够自动注入;-->
<!-- base-package:指定mapper接口的包名-->
```

```
<mybatis-spring:scan base-package="flipped.dao"/>

<bean id="mapper" class="org.mybatis.spring.mapper.MapperScannerConfigurer">
<property name="basePackage" value="flipped.dao"></property>
</bean>
```

4、理解Spring Bean

Spring属性注入的两种主要方式：

set方法注入、构造函数注入

Spring依赖注入（DI）的三种主要方式：

构造方法注入（Construct注入），setter注入，基于注解的注入（接口注入）

> DI和IoC是一样的, 当一个类(A)中需要依赖另一个类(B)对象时,把B赋值给A的过程就叫做依赖注入.

Bean：

1. bean是对象，一个或者多个不限定
2. bean由Spring中一个叫IoC的东西管理
3. 我们的应用程序由一个个bean构成

> https://www.awaimai.com/2596.html

5、SSM整合问题解决：

- **Access denied for user 'zh'@'localhost' (using password: YES)**

数据库连接属性（username）冲突，dbconfig.properties中属性命名尽量写成：jdbc.username，防止冲突

> https://blog.csdn.net/cnds123321/article/details/103915418

- **BindingException: Invalid bound statement (not found): flipped.dao.AccountMapper.getAccountInfo**

sql语句无法执行，反正就是检查 mapper文件的扫描配置以及mapper.xml中的namespace

> https://blog.csdn.net/benben513624/article/details/81076182
> 只有source文件夹下才能通过.创建多级文件目录

- **service层无法注入**

spring-servlet.xml 与spring配置文件中的扫描项目写mapper/service的上层目录

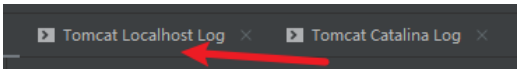
```
<context:component-scan base-package="flipped">
```

```
<context:component-scan base-package="flipped">
```

- **重点：通过Maven配置的项目需要注意，要手动将jar包添加到输出目录中去**

java.lang.ClassNotFoundException:org.springframework.web.context.ContextLoa

首先看tomcat日志



再导入jar包