

第三章 Maven项目管理

-Maven(三)2018/11/20 [泽林.王峰]

授课目标

- 1、上章回顾
- 2、Maven分模块开发（非常重要）-idea版本
- 3、私服（nexus）搭建（了解）

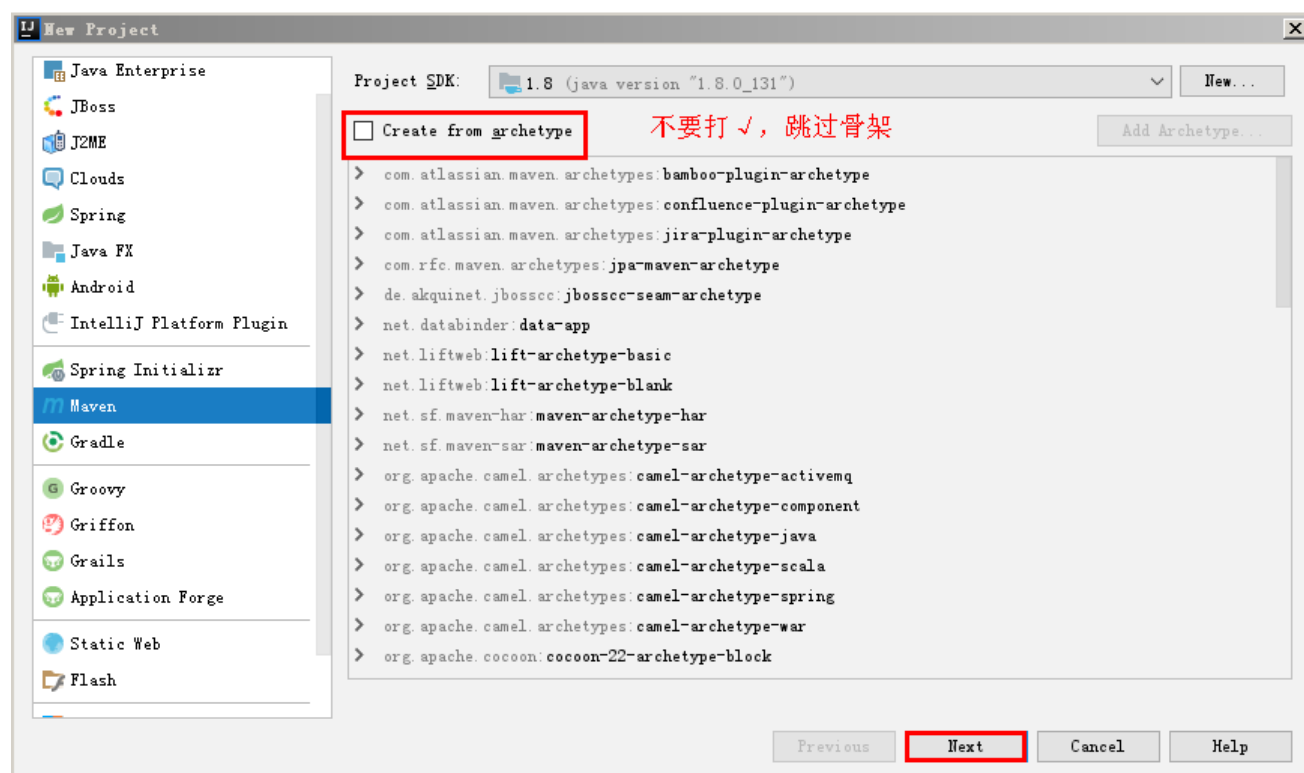
授课内容

1、 上章回顾

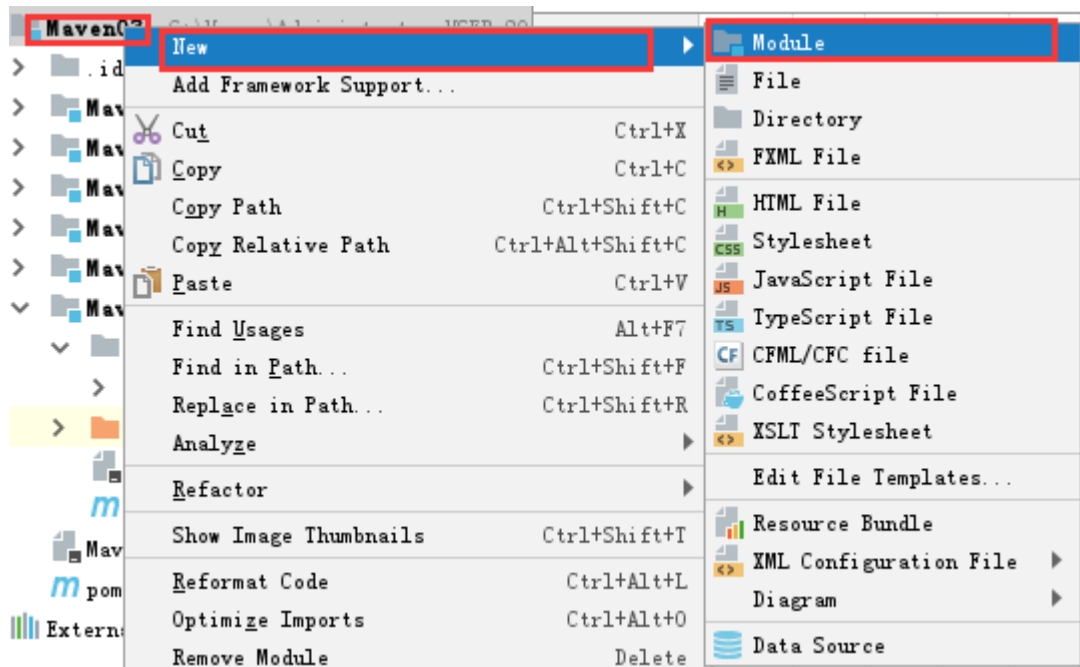
2、Maven分模块开发（非常重要）-idea版本

第一步：新建Maven工程及其下的各个模块：

1.1) 新建Maven工程（不选择骨架）



1.2) 在此工程下新建Maven-Pojo模块（与上面一样，选择跳转骨架）



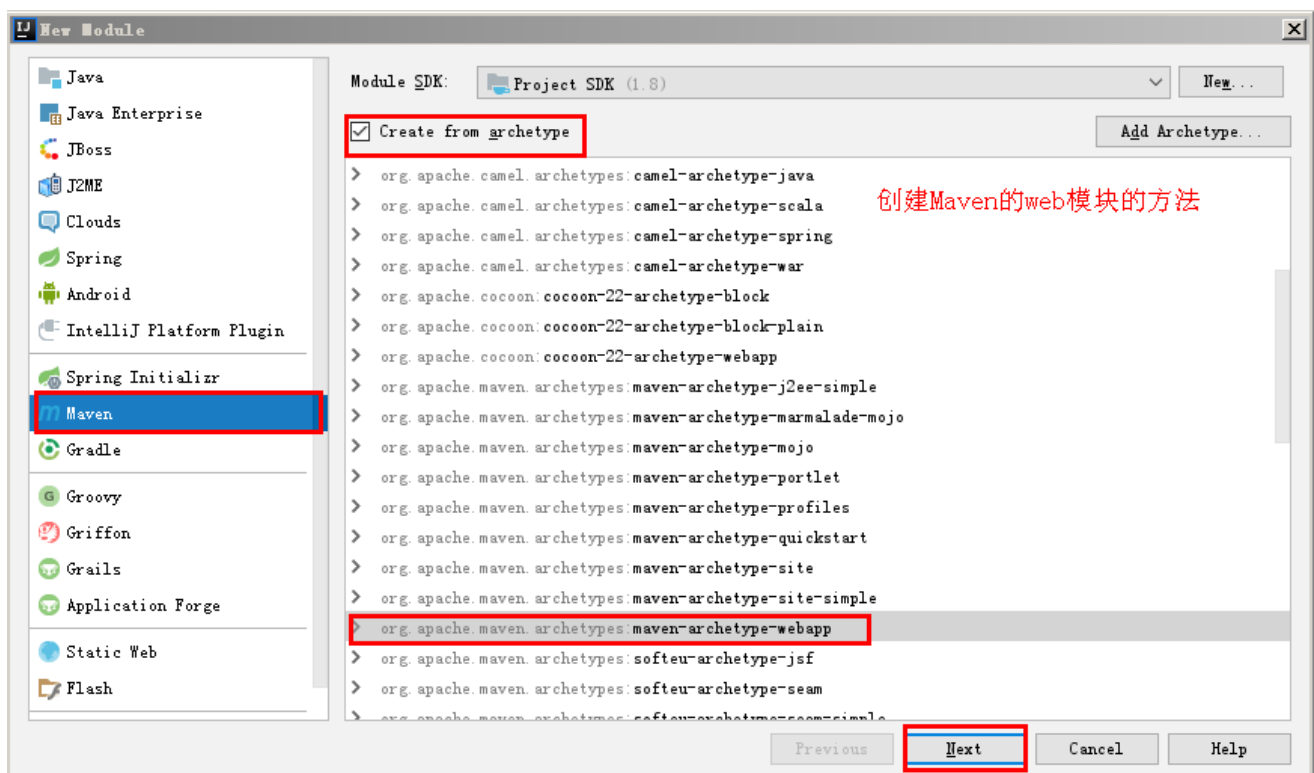
1.3) 在此工程下新建Maven-Interface模块 (与上面一样, 选择跳转骨架)

1.4) 在此工程下新建Maven-Dao模块 (与上面一样, 选择跳转骨架)

1.5) 在此工程下新建Maven-Service模块 (与上面一样, 选择跳转骨架)

1.6) 在此工程下新建Maven-Commons模块 (与上面一样, 选择跳转骨架)

1.7) 在此工程下新建Maven-Web模块



第二步 : 在各个模块之间建立依赖关系 :

2.1) 在Maven-Interface模块的pom.xml文件中添加对Maven-Pojo的依赖

```
1 <!-- 添加对pojo模块的依赖-->
2     <dependencies>
3         <dependency>
4             <groupId>com.zelin</groupId>
5             <artifactId>Maven_POJO</artifactId>
6             <version>0.0.1-SNAPSHOT</version>
7         </dependency>
8     <!-- 添加对Commons模块的依赖-->
9     <dependency>
10         <groupId>com.zelin</groupId>
11         <artifactId>Maven_Commons</artifactId>
12         <version>0.0.1-SNAPSHOT</version>
13     </dependency>
14 </dependencies>
```

2.2) 在Maven-Dao模块的pom.xml文件中添加对Maven-Pojo的依赖

```
1 <dependency>
2     <groupId>com.zelin</groupId>
3     <artifactId>Maven_POJO</artifactId>
4     <version>0.0.1-SNAPSHOT</version>
5 </dependency>
```

2.3) 在Maven-Service模块的pom.xml文件中添加对Maven-Interface及Maven-Dao的依赖

```
1 <!-- 引入服务接口模块的依赖 -->
2     <dependency>
3         <groupId>com.zelin</groupId>
4         <artifactId>Maven_Interface</artifactId>
5         <version>0.0.1-SNAPSHOT</version>
6     </dependency>
7 <!-- 引入Dao模块的依赖-->
8     <dependency>
9         <groupId>com.zelin</groupId>
10        <artifactId>Maven_DAO</artifactId>
11        <version>0.0.1-SNAPSHOT</version>
12    </dependency>
```

2.4) 在Maven-Web模块的pom.xml文件中添加对Maven-Service模块的依赖：

```
1 <dependency>
2     <groupId>com.zelin</groupId>
3     <artifactId>Maven_Service</artifactId>
4     <version>0.0.1-SNAPSHOT</version>
5 </dependency>
```

第三步：添加各个模块中对外部jar的依赖

3.1) 在Maven-Dao模块的pom.xml文件中依赖：(数据库访问相关)

```
1  【Maven-Dao的pom.xml】
2  <?xml version="1.0" encoding="UTF-8"?>
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
http://maven.apache.org/xsd/maven-4.0.0.xsd">
6      <parent>
7          <artifactId>Maven03</artifactId>
8          <groupId>com.zelin</groupId>
9          <version>0.0.1-SNAPSHOT</version>
10     </parent>
11     <modelVersion>4.0.0</modelVersion>
12
13     <artifactId>Maven_DAO</artifactId>
14
15     <dependencies>
16         <!-- 日志处理 -->
17         <dependency>
18             <groupId>org.slf4j</groupId>
19             <artifactId>slf4j-log4j12</artifactId>
20         </dependency>
21         <!-- Mybatis -->
22         <dependency>
23             <groupId>org.mybatis</groupId>
24             <artifactId>mybatis</artifactId>
25         </dependency>
26         <dependency>
27             <groupId>org.mybatis</groupId>
28             <artifactId>mybatis-spring</artifactId>
29         </dependency>
30         <dependency>
31             <groupId>com.github.miemiedev</groupId>
32             <artifactId>mybatis-paginator</artifactId>
33         </dependency>
34         <dependency>
35             <groupId>com.github.pagehelper</groupId>
36             <artifactId>pagehelper</artifactId>
37         </dependency>
38         <!-- MySql -->
39         <dependency>
40             <groupId>mysql</groupId>
41             <artifactId>mysql-connector-java</artifactId>
42         </dependency>
43         <!-- 连接池 -->
44         <dependency>
45             <groupId>com.alibaba</groupId>
46             <artifactId>druid</artifactId>
47         </dependency>
```

```

48     <dependency>
49         <groupId>com.zelin</groupId>
50         <artifactId>Maven_POJO</artifactId>
51         <version>0.0.1-SNAPSHOT</version>
52     </dependency>
53
54 </dependencies>
55 </project>

```

3.2) 在Maven-Service模块的pom.xml文件中依赖：(Spring访问相关)

```

1  【Maven-Service的pom.xml】：
2  <?xml version="1.0" encoding="UTF-8"?>
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
http://maven.apache.org/xsd/maven-4.0.0.xsd">
6      <parent>
7          <artifactId>Maven03</artifactId>
8          <groupId>com.zelin</groupId>
9          <version>0.0.1-SNAPSHOT</version>
10     </parent>
11     <modelVersion>4.0.0</modelVersion>
12
13     <artifactId>Maven_Service</artifactId>
14
15     <dependencies>
16         <!-- Jackson Json处理工具包 -->
17         <dependency>
18             <groupId>com.fasterxml.jackson.core</groupId>
19             <artifactId>jackson-databind</artifactId>
20         </dependency>
21         <!-- Spring -->
22         <dependency>
23             <groupId>org.springframework</groupId>
24             <artifactId>spring-context</artifactId>
25         </dependency>
26         <dependency>
27             <groupId>org.springframework</groupId>
28             <artifactId>spring-beans</artifactId>
29         </dependency>
30         <dependency>
31             <groupId>org.springframework</groupId>
32             <artifactId>spring-webmvc</artifactId>
33         </dependency>
34         <dependency>
35             <groupId>org.springframework</groupId>
36             <artifactId>spring-jdbc</artifactId>
37         </dependency>
38         <dependency>
39             <groupId>org.springframework</groupId>
40             <artifactId>spring-aspects</artifactId>

```

```

41     </dependency>
42     <dependency>
43         <groupId>org.springframework</groupId>
44         <artifactId>spring-jms</artifactId>
45     </dependency>
46     <dependency>
47         <groupId>org.springframework</groupId>
48         <artifactId>spring-context-support</artifactId>
49     </dependency>
50     <!-- 引入服务接口模块的依赖 -->
51     <dependency>
52         <groupId>com.zelin</groupId>
53         <artifactId>Maven_Interface</artifactId>
54         <version>0.0.1-SNAPSHOT</version>
55     </dependency>
56     <!--引入Dao模块的依赖-->
57     <dependency>
58         <groupId>com.zelin</groupId>
59         <artifactId>Maven_DAO</artifactId>
60         <version>0.0.1-SNAPSHOT</version>
61     </dependency>
62
63 </dependencies>
64 </project>

```

3.3) 在Maven-Web模块的pom.xml文件中依赖：

```

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      <parent>
8          <artifactId>Maven03</artifactId>
9          <groupId>com.zelin</groupId>
10         <version>0.0.1-SNAPSHOT</version>
11     </parent>
12     <modelVersion>4.0.0</modelVersion>
13
14     <artifactId>Maven_Web</artifactId>
15     <packaging>war</packaging>
16
17     <properties>
18         <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
19         <maven.compiler.source>1.7</maven.compiler.source>
20         <maven.compiler.target>1.7</maven.compiler.target>
21     </properties>
22
23     <dependencies>
24         <dependency>
25
26             <groupId>junit</groupId>

```

```

24         <artifactId>junit</artifactId>
25         <version>4.11</version>
26         <scope>test</scope>
27     </dependency>
28     <dependency>
29         <groupId>com.zelin</groupId>
30         <artifactId>Maven_Service</artifactId>
31         <version>0.0.1-SNAPSHOT</version>
32     </dependency>
33 </dependencies>
34
35 <build>
36     <finalName>Maven_Web</finalName>
37     <pluginManagement><!-- lock down plugins versions to avoid using Maven
defaults (may be moved to parent pom) -->
38         <plugins>
39             <plugin>
40                 <artifactId>maven-clean-plugin</artifactId>
41                 <version>3.0.0</version>
42             </plugin>
43             <!-- see http://maven.apache.org/ref/current/maven-core/default-
bindings.html#Plugin_bindings_for_war_packaging -->
44             <plugin>
45                 <artifactId>maven-resources-plugin</artifactId>
46                 <version>3.0.2</version>
47             </plugin>
48             <plugin>
49                 <artifactId>maven-compiler-plugin</artifactId>
50                 <version>3.7.0</version>
51             </plugin>
52             <plugin>
53                 <artifactId>maven-surefire-plugin</artifactId>
54                 <version>2.20.1</version>
55             </plugin>
56             <plugin>
57                 <artifactId>maven-war-plugin</artifactId>
58                 <version>3.2.0</version>
59             </plugin>
60             <plugin>
61                 <artifactId>maven-install-plugin</artifactId>
62                 <version>2.5.2</version>
63             </plugin>
64             <plugin>
65                 <artifactId>maven-deploy-plugin</artifactId>
66                 <version>2.8.2</version>
67             </plugin>
68
69             <!-- 资源文件拷贝插件 -->
70             <plugin>
71                 <groupId>org.apache.maven.plugins</groupId>
72                 <artifactId>maven-resources-plugin</artifactId>
73                 <version>2.7</version>
74
75         </plugins>
76     </pluginManagement>
77 </build>
78
79 <configuration>

```

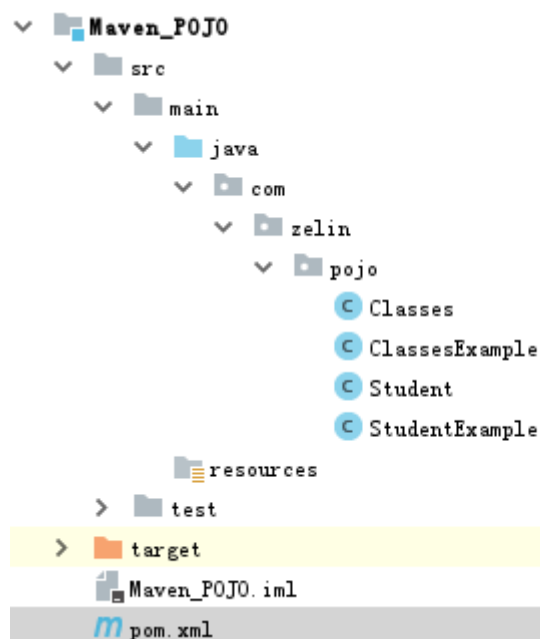
```

75         <encoding>UTF-8</encoding>
76     </configuration>
77 </plugin>
78 <!-- 配置Tomcat插件 -->
79 <plugin>
80     <groupId>org.apache.tomcat.maven</groupId>
81     <artifactId>tomcat7-maven-plugin</artifactId>
82     <version>2.2</version>
83     <configuration>
84         <port>9000</port>
85         <path>/</path>
86         <contextReloadable>true</contextReloadable>
87     </configuration>
88 </plugin>
89 </plugins>
90 </pluginManagement>
91 </build>
92 </project>
93

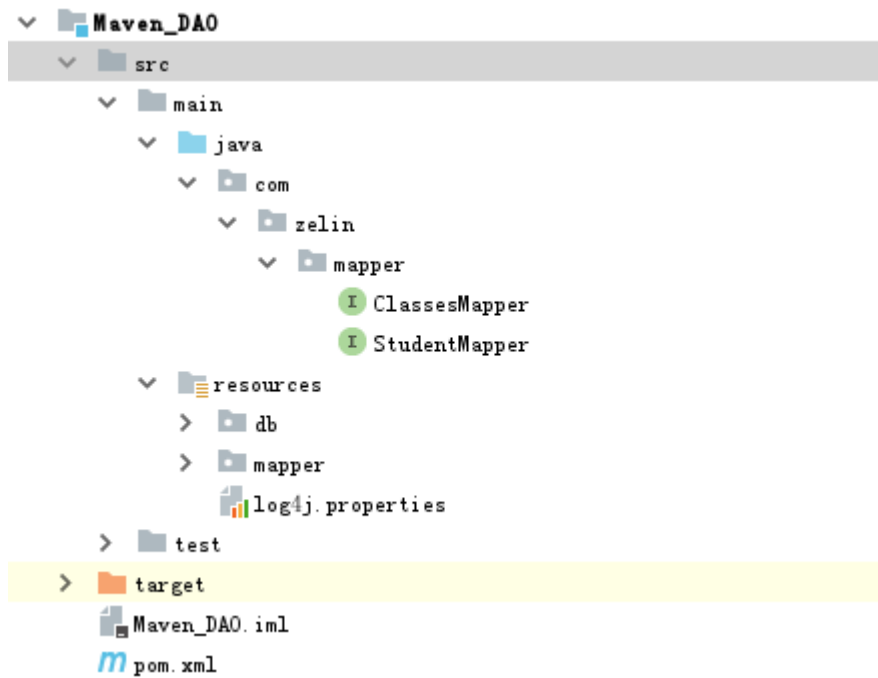
```

第四步：复制MyBatis自动生成相关的资源及自定义的配置文件

4.1) 复制MyBatis自动生成的pojo类到Maven-Pojo模块下：



4.2) 复制MyBatis自动生成的interface接口及自定义配置到MavenDao模块下：



注意：

1. 上图中的文件夹db中存放的是数据库访问相关字符串资源。内容大致如下：

```
1 db.driver=com.mysql.jdbc.Driver
2 db.url=jdbc:mysql:///java1301?useUnicode=true&characterEncoding=utf-8
3 db.user=root
4 db.password=123
```

2. 生成的Mapper接口如StudentMapper.java如下：

```
1 public interface StudentMapper {
2     long countByExample(StudentExample example);
3
4     int deleteByExample(StudentExample example);
5
6     int deleteByPrimaryKey(Integer sid);
7
8     int insert(Student record);
9
10    int insertSelective(Student record);
11
12    List<Student> selectByExample(StudentExample example);
13
14    Student selectByPrimaryKey(Integer sid);
15
16    int updateByExampleSelective(@Param("record") Student record,
17                               @Param("example") StudentExample example);
18
19    int updateByExample(@Param("record") Student record, @Param("example")
20                       StudentExample example);
21 }
```

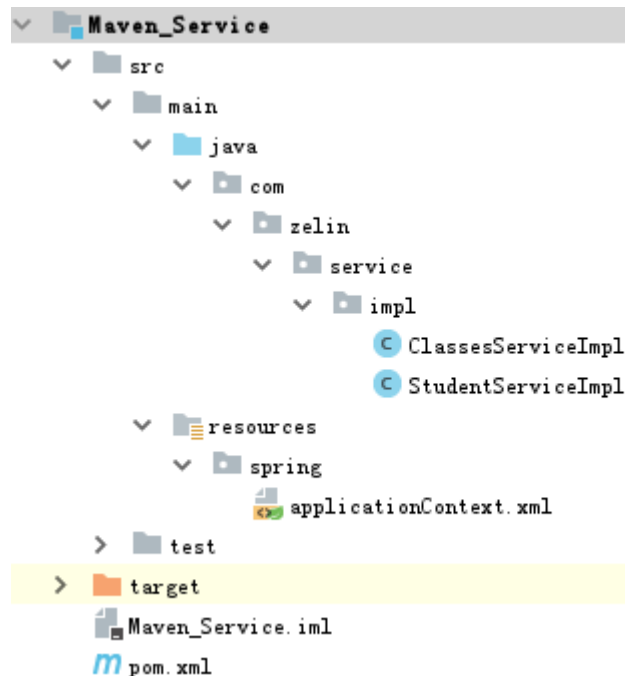
```

20     int updateByPrimaryKeySelective(Student record);
21
22     int updateByPrimaryKey(Student record);
23 }

```

3. 自动生成的Mapper映射文件略过。

4.3) 在Maven-Service模块定义配置文件及Service的实现，效果如下：



注意：其中的配置文件applicationContext.xml文件的内容大致如下：

```

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      xsi:schemaLocation="http://www.springframework.org/schema/beans
6          http://www.springframework.org/schema/beans/spring-beans.xsd
7          http://www.springframework.org/schema/context
8          http://www.springframework.org/schema/context/spring-context-4.2.xsd">
9
10     <!-- 读取属性文件 -->
11     <context:property-placeholder location="classpath*:db/db.properties"/>
12     <!-- 配置包扫描器 -->
13     <context:component-scan base-package="com.zelin.service.impl"/>
14     <!-- 配置数据源 -->
15     <bean id="dataSource" class="com.alibaba.druid.pool.DruidDataSource">
16         <property name="driverClassName" value="${db.driver}"/>
17         <property name="url" value="${db.url}"/>
18         <property name="username" value="${db.user}"/>
19         <property name="password" value="${db.password}"/>
20     </bean>
21     <!-- 配置SqlSessionFactoryBean -->
22     <bean id="sqlSessionFactoryBean"

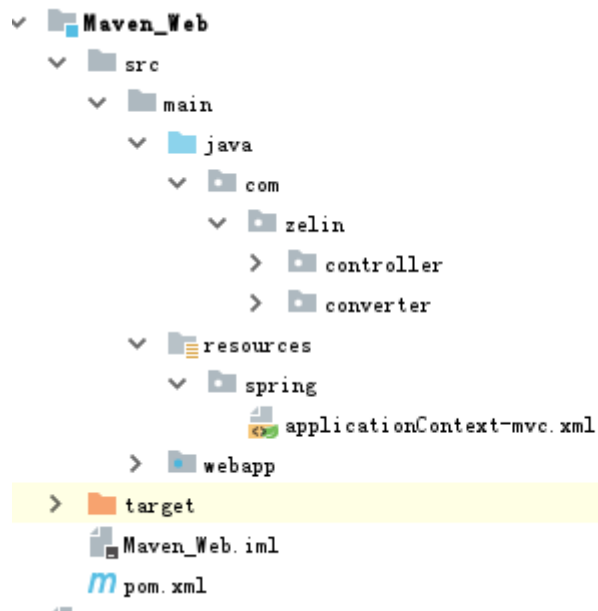
```

```

class="org.mybatis.spring.SqlSessionFactoryBean">
21     <property name="dataSource" ref="dataSource"/>
22     <property name="typeAliasesPackage" value="com.zelin.pojo"/>
23     <property name="plugins">
24         <list>
25             <bean class="com.github.pagehelper.PageHelper">
26                 <property name="properties">
27                     <value>
28                         dialect=mysql
29                     </value>
30                 </property>
31             </bean>
32         </list>
33     </property>
34     <property name="mapperLocations" value="classpath*:mapper/*.xml"/>
35 </bean>
36 <bean class="org.mybatis.spring.mapper.MapperScannerConfigurer">
37     <property name="basePackage" value="com.zelin.mapper"/>
38 </bean>
39 </beans>
40

```

4.4) 在Maven-Web模块定义配置文件及Controller的实现，效果如下：



注意：

1. 其中的applicationContext-mvc.xml文件的内容大致如下：

```

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:mvc="http://www.springframework.org/schema/mvc"
6     xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd
http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context-4.2.xsd
http://www.springframework.org/schema/mvc
http://www.springframework.org/schema/mvc/spring-mvc-4.2.xsd">
7
8     <context:component-scan base-package="com.zelin.controller"/>
9     <mvc:annotation-driven/>
10
11 </beans>

```

2. 其中的web.xml文件的内容大致如下：

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"
3     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4     xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd"
5     version="3.1">
6     <display-name>Maven_Web</display-name>
7     <!-- 1.加载spring所有的配置文件（不包含springmvc的配置文件） -->
8     <context-param>
9         <param-name>contextConfigLocation</param-name>
10        <!-- 注意：
11            classpath与classpath*区别：
12            (1) classpath:代表访问指定类路径下的资源文件
13            (2) classpath*:代表访问指定类路径及引入的第三方包中的资源文件。
14        -->
15        <param-value>classpath*:spring/applicationContext*.xml</param-value>
16    </context-param>
17    <!-- 2.配置spring的监听器 -->
18    <listener>    <listener-
class>org.springframework.web.context.ContextLoaderListener</listener-class>
19    </listener>
20
21    <!-- 3.配置DispatcherServlet -->
22    <servlet>
23        <servlet-name>springmvc</servlet-name>
24        <servlet-
class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
25        <init-param>
26            <param-name>contextConfigLocation</param-name>
27            <param-value>classpath:spring/applicationContext-mvc.xml</param-
value>
28        </init-param>
29    </servlet>
30    <servlet-mapping>
31        <servlet-name>springmvc</servlet-name>

```

```

32     <url-pattern>*.action</url-pattern>
33 </servlet-mapping>
34
35 <!-- 4.处理中文乱码的过滤器 -->
36 <filter>
37     <filter-name>characterEncoding</filter-name>
38     <filter-
class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>
39     <init-param>
40         <param-name>encoding</param-name>
41         <param-value>UTF-8</param-value>
42     </init-param>
43 </filter>
44 <filter-mapping>
45     <filter-name>characterEncoding</filter-name>
46     <url-pattern>/*</url-pattern>
47 </filter-mapping>
48 <welcome-file-list>
49     <welcome-file>index.html</welcome-file>
50     <welcome-file>index.htm</welcome-file>
51     <welcome-file>index.jsp</welcome-file>
52     <welcome-file>default.html</welcome-file>
53     <welcome-file>default.htm</welcome-file>
54     <welcome-file>default.jsp</welcome-file>
55 </welcome-file-list>
56 </web-app>

```

4.5) 最终运行效果如下：

学生管理系统							
学号	姓名	性别	年龄	住址	生日	所在班级	操作
2	王小五	男	28	广州	2010-10-27	1301班	修改 删除
7	罗成	男	22	邵阳	1997-1-18	1301班	修改 删除
8	魏征	男	28	洛ab阳	2000-3-16	1301班	修改 删除
15	赵本山	男	65	东北大街	1954-11-17	1301班	修改 删除
19	常遇春	男	100	河南	1918-8-28	1301班	修改 删除

« 1 2 » 第 1 页 每页 5 条 /共9条


添加学生 输入姓名关键字 输入住址关键字 请选择班级： 1301班 查询

泽林信息版权所有 2000-2018.

3、 私服 (nexus) 搭建 (了解)

4.1) 安装私服nexus：

将

 nexus-2.12.0-01-bundle.zip

解压到指定的目录中，然后，进入目录中的/bin目录

4.1.1) 开始安装私服：

Nexus.bat install

4.1.2) 启动服务：

Nexus.bat start

4.1.3) 如果想卸载可使用命令：

Nexus.bat uninstall

4.1.3) 找到私服安装目录下的conf文件夹，找到nexus.properties

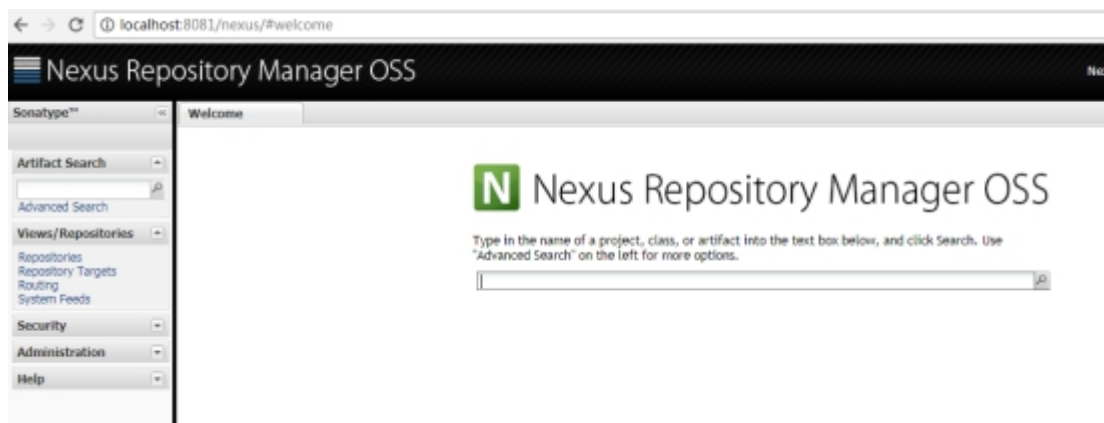
```
# Jetty section
application-port=8081
application-host=0.0.0.0
nexus-webapp=${bundleBasedir}/nexus
nexus-webapp-context-path=/nexus

# Nexus section
nexus-work=${bundleBasedir}/../sonatype-work/nexus
runtime=${bundleBasedir}/nexus/WEB-INF
```

端口号

上下文路径

4.1.4) 使用username:admin password:admin123这种方式登录私服。



4.2) 在maven的配置文件settings.xml文件中：

【第一部分】

```

1 <server>
2     <id>releases</id>
3     <username>admin</username>
4     <password>admin123</password>
5 </server>
6 <server>
7     <id>snapshots</id>
8     <username>admin</username>
9     <password>admin123</password>
10 </server>

```

【第二部分】：在profiles标签下配置

```

1 <profile>
2     <!--profile的id-->
3     <id>dev</id>
4     <repositories>
5         <repository>
6             <!--仓库id, repositories可以配置多个仓库, 保证id不重复-->
7             <id>nexus</id>
8             <!--仓库地址, 即nexus仓库组的地址-->
9             <url>http://localhost:8081/nexus/content/groups/public/</url>
10            <!--是否下载releases构件-->
11            <releases>
12                <enabled>true</enabled>
13            </releases>
14            <!--是否下载snapshots构件-->
15            <snapshots>
16                <enabled>true</enabled>
17            </snapshots>
18        </repository>
19    </repositories>
20    <pluginRepositories>
21        <!-- 插件仓库, maven的运行依赖插件, 也需要从私服下载插件 -->
22        <pluginRepository>
23            <!-- 插件仓库的id不允许重复, 如果重复后边配置会覆盖前边 -->
24            <id>public</id>
25            <name>Public Repositories</name>
26            <url>http://localhost:8081/nexus/content/groups/public/</url>
27        </pluginRepository>
28    </pluginRepositories>
29 </profile>
30

```

【第三部分】：在profiles标签外面激活上面的profile配置信息:

```

1 <activeProfiles>
2     <activeProfile>dev</activeProfile>
3 </activeProfiles>

```

4.3) 在Maven-D这个模块的pom.xml文件中添加如下配置：

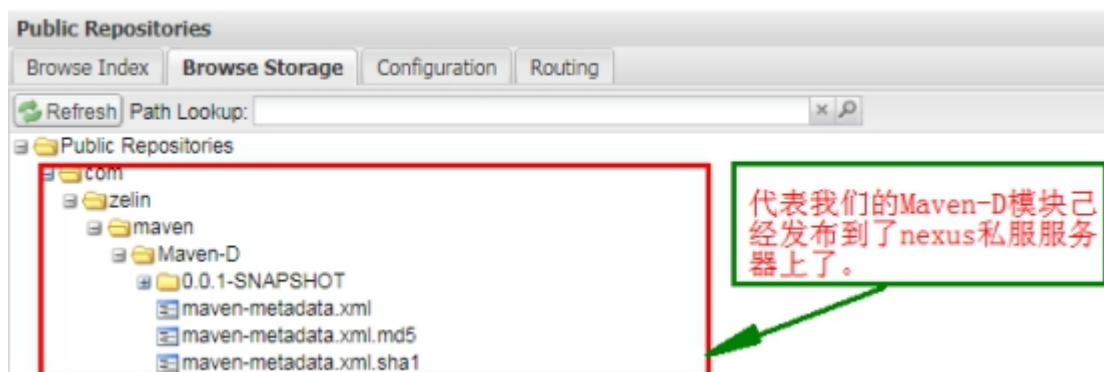
```
1 <distributionManagement>
2   <repository>
3     <id>releases</id>
4     <url>http://localhost:8081/nexus/content/repositories/releases/</url>
5   </repository>
6   <snapshotRepository>
7     <id>snapshots</id>
8     <url>http://localhost:8081/nexus/content/repositories/snapshots/</url>
9   </snapshotRepository>
10 </distributionManagement>
```

4.4) 将Maven-D deploy到nexus上，然后，再从本地仓库中删除Maven-D相关内容，最后，关闭Maven-D模块。

运行deploy命令后的效果如下：

```
--- maven-deploy-plugin:2.7:deploy (default-deploy) @ Maven-Dao ---
Downloading: http://localhost:8081/nexus/content/groups/public/org/codehaus/plexus/plexus-utils/1.
Downloaded: http://localhost:8081/nexus/content/groups/public/org/codehaus/plexus/plexus-utils/1.5
Downloading: http://localhost:8081/nexus/content/repositories/snapshots/com/zelin/Maven-Dao/0.0.1-
Uploading: http://localhost:8081/nexus/content/repositories/snapshots/com/zelin/Maven-Dao/0.0.1-SN
Uploaded: http://localhost:8081/nexus/content/repositories/snapshots/com/zelin/Maven-Dao/0.0.1-SNA
Uploading: http://localhost:8081/nexus/content/repositories/snapshots/com/zelin/Maven-Dao/0.0.1-SN
Uploaded: http://localhost:8081/nexus/content/repositories/snapshots/com/zelin/Maven-Dao/0.0.1-SNA
Downloading: http://localhost:8081/nexus/content/repositories/snapshots/com/zelin/Maven-Dao/maven-
Uploading: http://localhost:8081/nexus/content/repositories/snapshots/com/zelin/Maven-Dao/maven-
Uploaded: http://localhost:8081/nexus/content/repositories/snapshots/com/zelin/Maven-Dao/maven-me
Uploaded: http://localhost:8081/nexus/content/repositories/snapshots/com/zelin/Maven-Dao/maven-me
-----
```

在nexus服务器中查看效果如下：



4.5) 运行当前的Maven-Web，效果如下：

```
[INFO] Maven-P
[INFO] Maven-D
[INFO] Maven-S
[INFO] Maven-W
[INFO] Downloading: http://localhost:8081/nexus/content/groups/public/org/codehaus/mojo/maven-metadata.xml
[INFO] Downloaded: http://localhost:8081/nexus/content/groups/public/org/codehaus/mojo/maven-metadata.xml (20 KB at
[INFO] Downloaded: http://localhost:8081/nexus/content/groups/public/org/apache/maven/plugins/maven-metadata.xml (1
[INFO] Downloading: http://localhost:8081/nexus/content/groups/public/org/codehaus/mojo/tomcat-maven-plugin/maven-m
[INFO] Downloaded: http://localhost:8081/nexus/content/groups/public/org/codehaus/mojo/tomcat-maven-plugin/maven-me
[INFO] -----
```

A green arrow points from a text box to the 'Maven-D' directory in the log output. The text box contains the following text:

代表正在从私服服务器上下载我们使用的Maven-D模块

4.6) 程序最终运行效果同上。
