2048（一）

ssm项目

pom.xml文件

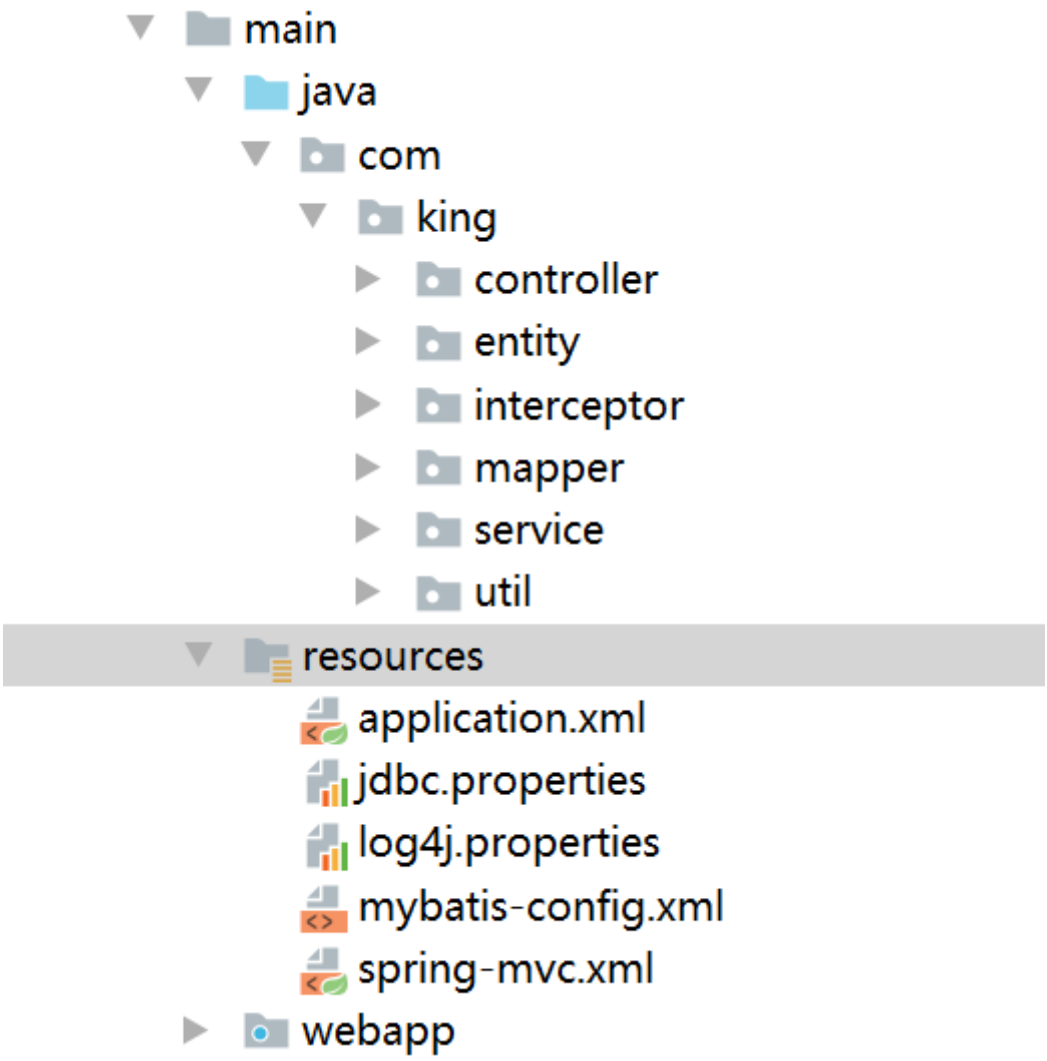
```
<dependencies>
    <dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>4.11</version>
        <scope>test</scope>
    </dependency>
    <!-- 引入 MyBatis + Spring jar-->
    <!-- spring基础包-->
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-context</artifactId>
        <version>5.1.2.RELEASE</version>
    </dependency>
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-beans</artifactId>
        <version>5.1.2.RELEASE</version>
    </dependency>
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-expression</artifactId>
        <version>5.1.2.RELEASE</version>
    </dependency>
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-core</artifactId>
        <version>5.1.2.RELEASE</version>
    </dependency>
    <!-- 2、SpringAOP包-->
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-aop</artifactId>
        <version>5.1.2.RELEASE</version>
    </dependency>
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-aspects</artifactId>
        <version>5.1.2.RELEASE</version>
    </dependency>
    <dependency>
        <groupId>org.aspectj</groupId>
        <artifactId>aspectjweaver</artifactId>
        <version>1.8.13</version>
    </dependency>
    <!--3、Spring事务包-->
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-tx</artifactId>
```

```
        <version>5.1.2.RELEASE</version>
    </dependency>
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-orm</artifactId>
        <version>5.1.2.RELEASE</version>
    </dependency>
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-oxm</artifactId>
        <version>5.1.2.RELEASE</version>
    </dependency>
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-jdbc</artifactId>
        <version>5.1.2.RELEASE</version>
    </dependency>
    <!--4、Spring的测试包-->
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-test</artifactId>
        <version>5.1.2.RELEASE</version>
    </dependency>
    <!--5、MyBatis包-->
    <dependency>
        <groupId>org.mybatis</groupId>
        <artifactId>mybatis</artifactId>
        <version>3.4.6</version>
    </dependency>
    <!--6、Mybatis和spring的插件包-->
    <dependency>
        <groupId>org.mybatis</groupId>
        <artifactId>mybatis-spring</artifactId>
        <version>1.3.2</version>
    </dependency>
    <!--7、数据库包-->
    <dependency>
        <groupId>mysql</groupId>
        <artifactId>mysql-connector-java</artifactId>
        <version>8.0.13</version>
    </dependency>
    <!--8、数据源包-->
    <dependency>
        <groupId>com.alibaba</groupId>
        <artifactId>druid</artifactId>
        <version>1.0.9</version>
    </dependency>
    <!--log4j-->
    <dependency>
        <groupId>log4j</groupId>
        <artifactId>log4j</artifactId>
        <version> 1.2.17</version>
    </dependency>
    <!-- spring mvc-->
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-webmvc</artifactId>
        <version>5.1.2.RELEASE</version>
```

```
    </dependency>
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-web</artifactId>
        <version>5.1.2.RELEASE</version>
    </dependency>
    <!-- 引入servlet的api-->
    <dependency>
        <groupId>javax.servlet</groupId>
        <artifactId>javax.servlet-api</artifactId>
        <version>3.0.1</version>
        <scope>provided</scope>
    </dependency>
    <dependency>
        <groupId>org.projectlombok</groupId>
        <artifactId>lombok</artifactId>
        <version>1.16.10</version>
    </dependency>
    <!--导入json依赖-->
    <dependency>
        <groupId>com.fasterxml.jackson.core</groupId>
        <artifactId>jackson-core</artifactId>
        <version>2.9.7</version>
    </dependency>
    <dependency>
        <groupId>com.fasterxml.jackson.core</groupId>
        <artifactId>jackson-databind</artifactId>
        <version>2.9.7</version>
    </dependency>
    <dependency>
        <groupId>com.fasterxml.jackson.core</groupId>
        <artifactId>jackson-annotations</artifactId>
        <version>2.9.0</version>
    </dependency>
    <!-- 文件上传-->
    <dependency>
        <groupId>commons-fileupload</groupId>
        <artifactId>commons-fileupload</artifactId>
        <version>1.3.3</version>
    </dependency>
    <dependency>
        <groupId>commons-io</groupId>
        <artifactId>commons-io</artifactId>
        <version>2.6</version>
    </dependency>
</dependencies>
<build>
    <plugins>
        <plugin>
            <groupId>org.apache.tomcat.maven</groupId>
            <artifactId>tomcat7-maven-plugin</artifactId>
            <version>2.2</version>
            <configuration>
                <port>80</port>
                <path>/</path>
            </configuration>
        </plugin>
    </plugins>
```

```
</build>
```

创建包及文件



entity下创建 User.java

```java
package com.king.entity;
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;
@Data
@AllArgsConstructor
@NoArgsConstructor
//@Data ： 注在类上，提供类的get、set、equals、hashCode、canEqual、toString方法
//@AllArgsConstructor ： 注在类上，提供类的全参构造
//@NoArgsConstructor ： 注在类上，提供类的无参构造
public class User {
  private long uid;
  private String uname;
  private String eMail;
  private String pass;
  private String state;
  private long age;
  private String sex;
}
```

util下创建工具类ResponseResult.java

```
package com.king.util;
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;
@Data
@AllArgsConstructor
@NoArgsConstructor
public class ResponseResult {
    private int code;  // layui中 0 正常 -1异常
    private int  count ;  // 结果集总记录数
    private Object data ;
    private String message;
    public ResponseResult(int code , String message){
        this.code = code ;
        this.message = message;
    }
}
```

mapper下创建一个UserMapper.java

```
package com.king.mapper;
import com.king.entity.User;
import org.apache.ibatis.annotations.Select;
public interface UserMapper {
    @Select("select * from user where e_mail = #{name}")
    //根据用户名查询
    public User getUser(String name);
}
```

service下创建UserService.java

```
package com.king.service;
import com.king.util.ResponseResult;
public interface UserService {
//登录
    ResponseResult login(String name, String pass);;
}
```

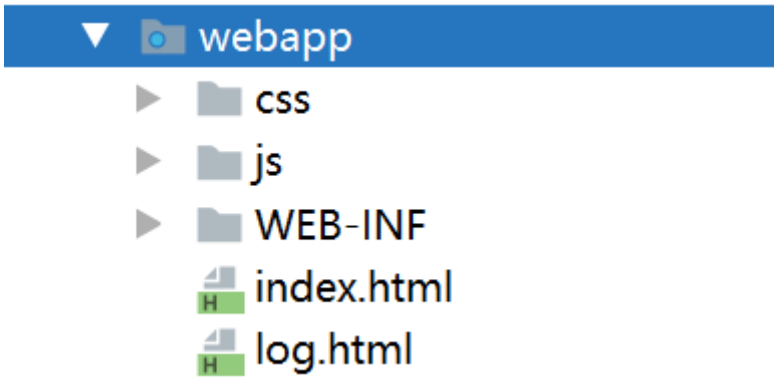
service下再创建一个包impl，并在impl包下创建一个 UserServiceImpl.java文件

```
package com.king.service.impl;
import com.king.entity.User;
import com.king.mapper.UserMapper;
import com.king.service.UserService;
import com.king.util.ResponseResult;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
@Service
public class UserServiceImpl implements UserService {
    @Autowired
    private UserMapper userMapper;
```

```
    @Override
    public ResponseResult login(String name, String pass) {
        ResponseResult result = new ResponseResult();
        User user = userMapper.getUser(name);
        userMapper.getUser(name);
        if (user!=null){
            if (user.getPass().equals(pass)){
                result.setCode(0);
                result.setMessage("登录成功");
                result.setData(user);
            }else {
                result.setCode(-1);
                result.setMessage("密码错误");
            }
        }
        else {
            result.setCode(-1);
            result.setMessage("用户不存在");
        }
       return result;
    }
}
```

在controller下创建一个UserController.java

```
package com.king.controller;
import com.king.service.UserService;
import com.king.util.ResponseResult;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.ResponseBody;
import javax.servlet.http.HttpSession;
@RequestMapping("/user")
@Controller
public class UserController {
    @Autowired
    private UserService userService;
    @ResponseBody
    @RequestMapping("/login")
    public ResponseResult login(String username ,String password , HttpSession
session){
        ResponseResult result =  userService.login(username,password);
        if(result.getCode()==0){
            //写入session作用域
            session.setAttribute("adminUser",result.getData());
        }
        return result;
    }
}
```



WEB-INF下的web.xml

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

<web-app>
  <display-name>Archetype Created Web Application</display-name>

  <!--1 、配置 xml地址  在监听器加载该类时，需要读取 application.xml-->
  <context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>classpath*:application.xml</param-value>
  </context-param>

  <!--2、引入过滤器，设置编码格式-->
  <filter>
    <filter-name>characterEncodingFilter</filter-name>
    <filter-
class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>
    <!-- 设置编码格式-->
    <init-param>
      <param-name>encoding</param-name>
      <param-value>UTF-8</param-value>
    </init-param>
  </filter>
  <filter-mapping>
    <filter-name>characterEncodingFilter</filter-name>
    <url-pattern>/*</url-pattern>
  </filter-mapping>
  <!--3、 配置一个上下文监听器 ，用于启动Spring的上下文容器 （取代之前我们用main方法启动容
器）-->
  <listener>
    <listener-
class>org.springframework.web.context.ContextLoaderListener</listener-class>
  </listener>

  <!-- 4、引入Springmvc 核心处理类-->
  <servlet>
    <servlet-name>dispatcherServlet</servlet-name>
    <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-
class>
    <init-param>
      <param-name>contextConfigLocation</param-name>
      <param-value>classpath*:spring-mvc.xml</param-value>
    </init-param>
```

```
  </servlet>
  <servlet-mapping>
    <servlet-name>dispatcherServlet</servlet-name>
    <url-pattern>/</url-pattern>
  </servlet-mapping>

</web-app>
```