分布式架构搭建拉勾教育PC站

--- **老孙**

课程介绍：

1、项目架构

2、开发后端服务接口

3、前端门户系统联调

1、项目架构

1.1 项目介绍

- 拉勾教育PC站，是提供给我们学员观看技术视频的网站
- 学员使用手机号注册登录后，可以选择适合自己的课程，并观看课程视频，当然，有免费的课程，也有vip专属课程

1.2 页面原型展示

访问：http://edufront.lagou.com/

用户名：15510792995 密码：111111

1.3 技术选型

1.3.1 前端技术选型

技术名称	说明
Vue.js	是一套用于构建用户界面的渐进式JavaScript框架
Element UI库	element-ui 是饿了么前端出品的基于 Vue.js的 后台组件库，方便程序员进行页面快速布局和构建
node.js	简单的说 Node.js 就是运行在服务端的 JavaScript 运行环境
axios	对ajax的封装, 简单来说就是ajax技术实现了局部数据的刷新，axios实现了对ajax的封装

1.3.2 后端技术选型

技术名称	说明
Web层	借助springmvc接收请求，进行视图跳转
Service层	借助spring进行IOC、AOP、及事务管理
dao层	借助mybatis进行数据库交互
EasyCode插件	IDEA快速生成实体类的插件
Zookeeper	服务注册与服务发现
Dubbo	分布式框架，远程RPC调用
Redis	内存数据库，缓存
Lombok	消除实体类中冗余的get和set
SpringSocial	SpringSocial (Spring社交)，简单理解就是和第三方应用打交道，微信登录用

1.4 项目开发环境

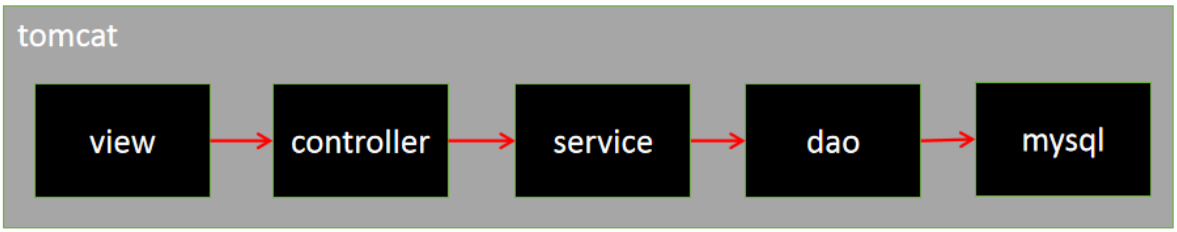
- 开发工具
  - 后端: IDEA 2019
  - 前端: VS code
  - 数据库客户端工具: SQLYog
- 开发环境
  - JDK 11
  - Maven 3.6.3
  - MySQL 5.7
  - Zookeeper 3.6.0
  - Dubbo 2.5.7
  - Redis 5.0.4

2、开发后端服务接口

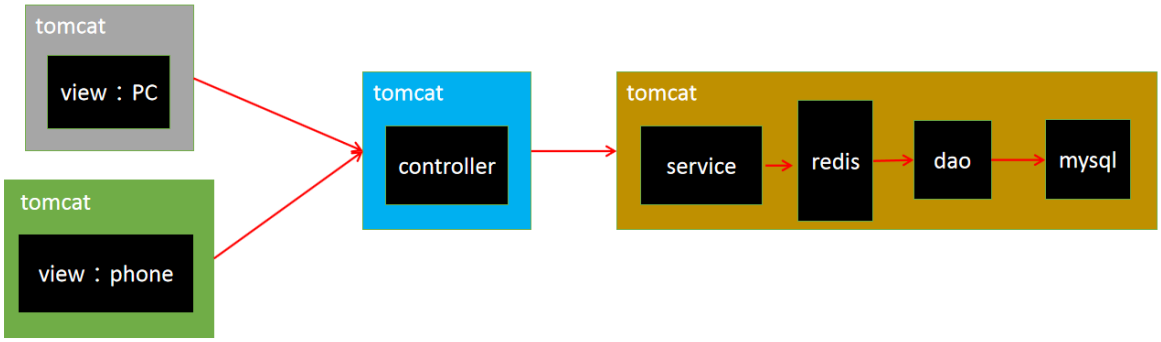
- 我们采用前后端分离的开发模式，先开发后端服务接口，测试成功，再开发前端vue界面，最后进行前后端联调，项目上线

2.1 项目结构与命名

- 单一架构：



- 分布式架构：



- 后端项目架构，我们采用dubbo的生产者和消费者的理论，创建服务提供方和服务消费方两个工程，通过maven聚合工程来搭建，模块划分如下：
  - 服务提供
    - lagou-edu-parent：pom聚合父工程，统一依赖设置
    - lagou-edu-entity：jar工程，封装实体类
    - lagou-edu-dao：jar工程，封装与数据库打交道的部分
    - lagou-edu-service：web工程，暴露服务的接口和实现
  - 服务消费
    - lagou-edu-web：web工程，接收前端工程发来的请求，远程调用服务并消费

2.1.2 URL命名

- 查询：http://localhost:8002/course/getList --- get开头
- 保存：http://localhost:8002/course/saveXx --- save开头
- 更新：http://localhost:8002/course/updateXx --- update开头

2.1.3 接口响应格式

```java
/**
  数据传输对象
 */
@Data
@AllArgsConstructor
@NoArgsConstructor
@ToString
public class ResponseDTO<T> implements Serializable {

    private static final long serialVersionUID = 1L;
    private int state;  // 操作状态
    private String message;  // 状态描述
    private T content;  // 相应内容
}
```

2.1.4 pom.xml

```xml
<properties>
    <spring.version>5.0.6.RELEASE</spring.version>
</properties>

<dependencies>
    <!-- Spring -->
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-context</artifactId>
        <version>${spring.version}</version>
    </dependency>
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-beans</artifactId>
        <version>${spring.version}</version>
    </dependency>
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-webmvc</artifactId>
        <version>${spring.version}</version>
    </dependency>
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-jdbc</artifactId>
        <version>${spring.version}</version>
    </dependency>
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-aspects</artifactId>
        <version>${spring.version}</version>
    </dependency>
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-test</artifactId>
        <version>${spring.version}</version>
        <scope>test</scope>
    </dependency>
    <!-- Mybatis -->
```

```
    <dependency>
        <groupId>org.mybatis</groupId>
        <artifactId>mybatis</artifactId>
        <version>3.2.8</version>
    </dependency>
    <dependency>
        <groupId>org.mybatis</groupId>
        <artifactId>mybatis-spring</artifactId>
        <version>1.3.2</version>
    </dependency>
    <!-- 连接池 -->
    <dependency>
        <groupId>com.alibaba</groupId>
        <artifactId>druid</artifactId>
        <version>1.0.9</version>
    </dependency>
    <!-- 数据库 -->
    <dependency>
        <groupId>mysql</groupId>
        <artifactId>mysql-connector-java</artifactId>
        <version>5.1.32</version>
    </dependency>
    <!--dubbo -->
    <dependency>
        <groupId>com.alibaba</groupId>
        <artifactId>dubbo</artifactId>
        <version>2.5.7</version>
    </dependency>
    <!--zookeeper -->
    <dependency>
        <groupId>org.apache.zookeeper</groupId>
        <artifactId>zookeeper</artifactId>
        <version>3.4.6</version>
    </dependency>
     <!--zookeeper客户端 -->
    <dependency>
        <groupId>com.github.sgroschupf</groupId>
        <artifactId>zkclient</artifactId>
        <version>0.1</version>
    </dependency>
    <dependency>
        <groupId>javassist</groupId>
        <artifactId>javassist</artifactId>
        <version>3.11.0.GA</version>
    </dependency>
    <!-- fastjson -->
    <dependency>
        <groupId>com.alibaba</groupId>
        <artifactId>fastjson</artifactId>
        <version>1.2.47</version>
    </dependency>
    <!-- junit -->
    <dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>4.12</version>
        <scope>test</scope>
    </dependency>
```

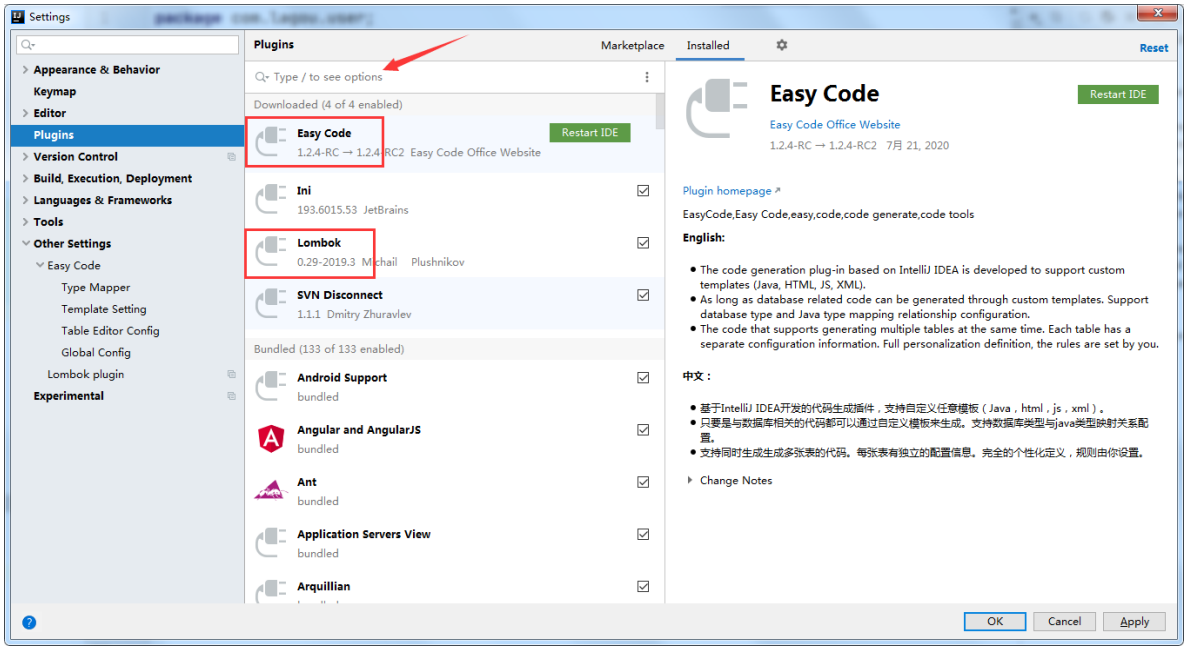
```
    <!--spring操作redis的工具类-->
    <dependency>
        <groupId>org.springframework.data</groupId>
        <artifactId>spring-data-redis</artifactId>
        <version>2.3.2.RELEASE</version>
    </dependency>
    <!--redis客户端-->
    <dependency>
        <groupId>redis.clients</groupId>
        <artifactId>jedis</artifactId>
        <version>3.1.0</version>
    </dependency>
    <!--json解析工具-->
    <dependency>
        <groupId>com.fasterxml.jackson.core</groupId>
        <artifactId>jackson-databind</artifactId>
        <version>2.9.8</version>
    </dependency>
</dependencies>
```

2.1.5 初始化数据库

- 数据库设计文档V1.0.pdf
- edu.sql

2.1 用户模块

- 实体类的编写没有任意技术含量，而且还浪费时间，传说中的“老牛活”
- 使用一个插件可以快速生成 entity，dao，service，mapper等文件
- 生成“老牛活”代码的解决方案有很多种：企业中比较常见的还有“mybatis的逆向工程（自学）”



2.1.1 用户登录/注册

功能名称	用户注册/登录
功能描述	用户输入手机号、密码，点击登录，调用后端接口开始登录 如果该手机号未注册，则自动注册并登录
功能接口	/user/login
备注	无

- **lagou-edu-entity**

lombok小辣椒，使用注解取代原来冗余的get和set，空构造，全参数构造

```
<!-- https://mvnrepository.com/artifact/org.projectlombok/lombok -->
<dependency>
    <groupId>org.projectlombok</groupId>
    <artifactId>lombok</artifactId>
    <version>1.18.12</version>
    <scope>provided</scope>
</dependency>
```

```
@Data  //get和set都全部生成了
@AllArgsConstructor //生成全参数的构造方法
@NoArgsConstructor //生成空构造方法
@ToString // 生成toString方法
public class User implements Serializable {
    private static final long serialVersionUID = -89788707895046947L;
    /**
    * 用户id
    */
    private Integer id;
```

- **lagou-edu-dao**

mybatis-config.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE configuration PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
"http://mybatis.org/dtd/mybatis-3-config.dtd">
<configuration>
    <!-- 后台的日志输出：针对开发者-->
    <settings>
        <setting name="logImpl" value="STDOUT_LOGGING"/>
    </settings>
</configuration>
```

spring-dao.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:tx="http://www.springframework.org/schema/tx"
        xmlns:context="http://www.springframework.org/schema/context"
        xsi:schemaLocation="
        http://www.springframework.org/schema/beans
        http://www.springframework.org/schema/beans/spring-beans.xsd
        http://www.springframework.org/schema/tx
        http://www.springframework.org/schema/tx/spring-tx.xsd
        http://www.springframework.org/schema/context
        http://www.springframework.org/schema/context/spring-context.xsd">
    <!--1.扫描包-->
    <context:component-scan base-package="mapper"/>
    <!--2.数据连接池-->
    <bean id="dataSource" class="com.alibaba.druid.pool.DruidDataSource">
        <property name="url" value="jdbc:mysql://192.168.204.141:3306/edu?
serverTimezone=GMT"/>
        <property name="username" value="root"/>
        <property name="password" value="123123"/>
        <property name="maxActive" value="10"/>
        <property name="minIdle" value="5"/>
    </bean>

    <!--3.sqlsessionFactory-->
    <bean id="sqlSessionFactory"
class="org.mybatis.spring.SqlSessionFactoryBean">
        <property name="dataSource" ref="dataSource"/>
        <property name="configLocation" value="classpath:mybatis/mybatis-
config.xml"/>
    </bean>
    <!--4.事务管理器-->
    <bean id="transactionManager"
class="org.springframework.jdbc.datasource.DataSourceTransactionManager">
        <property name="dataSource" ref="dataSource"/>
    </bean>
    <!--5.开启事务-->
    <tx:annotation-driven/>

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

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
"http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="mapper.UserDao">

    <resultMap type="com.lagou.entity.User" id="UserMap">
        <result property="id" column="id" jdbcType="INTEGER"/>
        <result property="name" column="name" jdbcType="VARCHAR"/>
        <result property="portrait" column="portrait" jdbcType="VARCHAR"/>
        <result property="phone" column="phone" jdbcType="VARCHAR"/>
        <result property="password" column="password" jdbcType="VARCHAR"/>
        <result property="regIp" column="reg_ip" jdbcType="VARCHAR"/>
        <result property="accountNonExpired" column="account_non_expired"
jdbcType="OTHER"/>
        <result property="credentialsNonExpired"
column="credentials_non_expired" jdbcType="OTHER"/>
```

```
        <result property="accountNonLocked" column="account_non_locked"
jdbcType="OTHER"/>
        <result property="status" column="status" jdbcType="VARCHAR"/>
        <result property="isDel" column="is_del" jdbcType="OTHER"/>
        <result property="createTime" column="create_time"
jdbcType="TIMESTAMP"/>
        <result property="updateTime" column="update_time"
jdbcType="TIMESTAMP"/>
    </resultMap>


    <select id="login" resultMap="UserMap">
        select * from user where phone = #{phone} and password = #{password}
    </select>

</mapper>
```

```
@Service
public interface UserDao {
   /**
    * @param phone 手机号
    * @param password 密码
    * @return 用户对象
    */
   User login(@Param("phone") String phone, @Param("password") String password);
}
```

```
// 测试类
@RunWith(SpringJUnit4ClassRunner.class)
@ContextConfiguration(locations = { "classpath:spring/spring-dao.xml" })
public class TestUser {

    @Autowired
    private UserDao userDao;

    @Test
    public void login(){
        User user = userDao.login("1101", "123");
        System.out.println("user = " + user);
    }
}
```

- **lagou-edu-service**

spring-service.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:dubbo="http://code.alibabatech.com/schema/dubbo"
       xsi:schemaLocation="
       http://www.springframework.org/schema/beans
       http://www.springframework.org/schema/beans/spring-beans.xsd
       http://code.alibabatech.com/schema/dubbo
```

```
        http://code.alibabatech.com/schema/dubbo/dubbo.xsd">

    <!--1.服务提供方在zookeeper中的“别名”-->
    <dubbo:application name="edu-service"/>
    <!--2.注册中心的地址-->
    <dubbo:registry address="zookeeper://192.168.204.141:2181"/>
    <!--3.扫描类（将什么包下的类作为服务提供类）-->
    <dubbo:annotation package="com.lagou"/>

    <dubbo:provider timeout="60000"/>

    <bean id="redisTemplate"
class="org.springframework.data.redis.core.RedisTemplate">
        <property name="connectionFactory" ref="connectionFactory"></property>
    </bean>
    <bean id="connectionFactory"
class="org.springframework.data.redis.connection.jedis.JedisConnectionFactory">
        <property name="hostName" value="192.168.204.141"></property>
        <property name="port" value="6379"/>
    </bean>
</beans>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xmlns="http://xmlns.jcp.org/xml/ns/javaee"
         xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd"
         id="WebApp_ID" version="3.1">
    <listener>
        <listener-
class>org.springframework.web.context.ContextLoaderListener</listener-class>
    </listener>
    <context-param>
        <param-name>contextConfigLocation</param-name>
        <param-value>classpath*:spring/spring-*.xml</param-value>
    </context-param>
</web-app>
```

```
// 测试类
@RunWith(SpringJUnit4ClassRunner.class)
@ContextConfiguration(locations = { "classpath*:spring/spring-*.xml" })
public class TestUser {

    @Autowired
    private UserService userService;

    @Test
    public void login(){
        User user = userService.login("110", "123");
        System.out.println("user = " + user);
    }
}
```

tomcat插件

```
<packaging>war</packaging>
```

```
<build>
    <plugins>
        <plugin>
            <groupId>org.apache.tomcat.maven</groupId>
            <artifactId>tomcat7-maven-plugin</artifactId>
            <configuration>
                <port>8001</port>
                <path>/</path>
            </configuration>
            <executions>
                <execution>
                    <!-- 打包完成后,运行服务 -->
                    <phase>package</phase>
                    <goals>
                        <goal>run</goal>
                    </goals>
                </execution>
            </executions>
        </plugin>
    </plugins>
</build>
```

- **lagou-edu-web**

spring-consumer.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:dubbo="http://code.alibabatech.com/schema/dubbo"
       xmlns:mvc="http://www.springframework.org/schema/mvc"
       xsi:schemaLocation="
       http://www.springframework.org/schema/beans
       http://www.springframework.org/schema/beans/spring-beans.xsd
       http://www.springframework.org/schema/mvc
       http://www.springframework.org/schema/mvc/spring-mvc.xsd
       http://code.alibabatech.com/schema/dubbo
       http://code.alibabatech.com/schema/dubbo/dubbo.xsd">

    <!--json转换器-->
    <mvc:annotation-driven>
        <mvc:message-converters register-defaults="true">
            <bean
class="com.alibaba.fastjson.support.spring.FastJsonHttpMessageConverter">
                <property name="supportedMediaTypes" value="application/json"/>
                <property name="features">
                    <array>
                        <value>WriteMapNullValue</value>
                        <value>WriteDateUseDateFormat</value>
                    </array>
                </property>
            </bean>
        </mvc:message-converters>
    </mvc:annotation-driven>
```

```
    <!--1.服务提供方在zookeeper中的“别名”-->
    <dubbo:application name="edu-web"/>
    <!--2.注册中心的地址-->
    <dubbo:registry address="zookeeper://192.168.204.141:2181"/>
    <!--3.扫描类（将什么包下的类作为消费类）-->
    <dubbo:annotation package="com.lagou"/>

</beans>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xmlns="http://xmlns.jcp.org/xml/ns/javaee"
         xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd"
         id="WebApp_ID" version="3.1">
    <!-- 解决post乱码 -->
    <filter>
        <filter-name>charset</filter-name>
        <filter-
class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>
        <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>charset</filter-name>
        <url-pattern>/*</url-pattern>
    </filter-mapping>

    <!--跨域配置-->
    <filter>
        <filter-name>corsFitler</filter-name>
        <filter-class>com.thetransactioncompany.cors.CORSFilter</filter-class>
    </filter>
    <filter-mapping>
        <filter-name>corsFitler</filter-name>
        <url-pattern>/*</url-pattern>
    </filter-mapping>

    <servlet>
        <servlet-name>springMVC</servlet-name>
        <servlet-
class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
        <init-param>
            <param-name>contextConfigLocation</param-name>
            <param-value>classpath:spring/spring-consumer.xml</param-value>
        </init-param>
    </servlet>
    <servlet-mapping>
        <servlet-name>springMVC</servlet-name>
        <url-pattern>/</url-pattern>
```

```
    </servlet-mapping>
</web-app>
```

tomcat插件

```
<packaging>war</packaging>

<build>
    <plugins>
        <plugin>
            <groupId>org.apache.tomcat.maven</groupId>
            <artifactId>tomcat7-maven-plugin</artifactId>
            <configuration>
                <port>8002</port>
                <path>/</path>
            </configuration>
            <executions>
                <execution>
                    <!-- 打包完成后,运行服务 -->
                    <phase>package</phase>
                    <goals>
                        <goal>run</goal>
                    </goals>
                </execution>
            </executions>
        </plugin>
    </plugins>
</build>
```

2.1.2 微信快捷登录

功能名称	用户注册/登录
功能描述	使用微信扫一扫，登录本系统
功能接口	/user/wxLogin

- **lagou-edu-entity**

```

```

- **lagou-edu-dao**

```

```

- **lagou-edu-service**

```

```

- **lagou-edu-web**

2.2 课程模块

2.2.1 全部课程

- 课程：Java从小白到大神（**course**） -------> 起点老师：拉勾网高级讲师（**teacher**）
  - 第一章：初识（**course_section**）
    - 01 什么是java（**course_lesson**）
    - 02 jdk的介绍与安装
  - 第二章：应用
    - 01 什么是数据类型
    - 02 什么是变量

course **1：1** teacher

course **1：N** course_section

course_section **1：N** course_lesson

course_lesson**1：N** course_media

功能名称	**选课**
功能描述	1、用户在未登录状态。获取所有上架的课程，课程顺序为： 配置活动信息标签的优先显示，再根据课程显示序列号顺序显示； 序列号相同按照创建时间倒序显示，课程都是未购买状态。
功能接口	/course/getAllCourse

- lagou-edu-entity

```
@Data
@AllArgsConstructor
@NoArgsConstructor
@ToString
public class Course implements Serializable {
    private Teacher teacher; // 一门课程对应一个讲师
    private List<CourseSection> courseSections; //一门课程对应多个章节
    private ActivityCourse activityCourse;// 一门课做一个活动
    // 省略
}
```

```
@Data
@AllArgsConstructor
@NoArgsConstructor
@ToString
public class CourseSection implements Serializable {
    private List<CourseLesson> courseLessons; // 一章对应多个小节
    // 省略
}
```

```
@Data
@AllArgsConstructor
@NoArgsConstructor
@ToString
public class CourseLesson implements Serializable {
    private CourseMedia courseMedia; // 一小节课对应一个视频
    // 省略
}
```

- lagou-edu-dao

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
"http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="mapper.CourseDao">

    <resultMap type="com.lagou.entity.Course" id="CourseMap">
        <result property="id" column="c_id" jdbcType="OTHER"/>
        <result property="courseName" column="course_name" jdbcType="VARCHAR"/>
        <result property="brief" column="brief" jdbcType="VARCHAR"/>
        <result property="price" column="price" jdbcType="OTHER"/>
        <result property="priceTag" column="price_tag" jdbcType="VARCHAR"/>
        <result property="discounts" column="discounts" jdbcType="OTHER"/>
        <result property="discountsTag" column="discounts_tag"
jdbcType="VARCHAR"/>
        <result property="courseDescriptionMarkDown"
column="course_description_mark_down" jdbcType="OTHER"/>
        <result property="courseDescription" column="course_description"
jdbcType="OTHER"/>
        <result property="courseImgUrl" column="course_img_url"
jdbcType="VARCHAR"/>
        <result property="isNew" column="is_new" jdbcType="OTHER"/>
        <result property="isNewDes" column="is_new_des" jdbcType="VARCHAR"/>
        <result property="lastOperatorId" column="last_operator_id"
jdbcType="INTEGER"/>
        <result property="autoOnlineTime" column="auto_online_time"
jdbcType="TIMESTAMP"/>
        <result property="createTime" column="c_create_time"
jdbcType="TIMESTAMP"/>
        <result property="updateTime" column="c_update_time"
jdbcType="TIMESTAMP"/>
        <result property="isDel" column="c_is_del" jdbcType="OTHER"/>
        <result property="totalDuration" column="total_duration"
jdbcType="INTEGER"/>
        <result property="courseListImg" column="course_list_img"
jdbcType="VARCHAR"/>
        <result property="status" column="c_status" jdbcType="INTEGER"/>
```

```
        <result property="sortNum" column="sort_num" jdbcType="INTEGER"/>
        <result property="previewFirstField" column="preview_first_field"
jdbcType="VARCHAR"/>
        <result property="previewSecondField" column="preview_second_field"
jdbcType="VARCHAR"/>
        <result property="sales" column="sales" jdbcType="INTEGER"/>

        <!-- 1：老师-->
        <association property="teacher" javaType="com.lagou.entity.Teacher">
            <result property="id" column="t_id" jdbcType="OTHER"/>
            <result property="courseId" column="t_course_id"
jdbcType="INTEGER"/>
            <result property="teacherName" column="teacher_name"
jdbcType="VARCHAR"/>
            <result property="position" column="position" jdbcType="VARCHAR"/>
            <result property="description" column="t_description"
jdbcType="VARCHAR"/>
            <result property="createTime" column="t_create_time"
jdbcType="TIMESTAMP"/>
            <result property="updateTime" column="t_update_time"
jdbcType="TIMESTAMP"/>
            <result property="isDel" column="t_is_del" jdbcType="OTHER"/>
        </association>

        <!-- 1：活动课程-->
        <association property="activityCourse"
javaType="com.lagou.entity.ActivityCourse">
            <result property="id" column="ac_id" jdbcType="INTEGER"/>
            <result property="courseId" column="ac_course_id"
jdbcType="INTEGER"/>
            <result property="beginTime" column="begin_time"
jdbcType="TIMESTAMP"/>
            <result property="endTime" column="end_time" jdbcType="TIMESTAMP"/>
            <result property="amount" column="amount" jdbcType="OTHER"/>
            <result property="stock" column="stock" jdbcType="INTEGER"/>
            <result property="status" column="ac_status" jdbcType="OTHER"/>
            <result property="isDel" column="ac_is_del" jdbcType="OTHER"/>
            <result property="remark" column="remark" jdbcType="VARCHAR"/>
            <result property="createTime" column="ac_create_time"
jdbcType="TIMESTAMP"/>
            <result property="createUser" column="create_user"
jdbcType="VARCHAR"/>
            <result property="updateTime" column="ac_update_time"
jdbcType="TIMESTAMP"/>
            <result property="updateUser" column="update_user"
jdbcType="VARCHAR"/>
        </association>
        <!--N：章节-->
        <collection property="courseSections"
ofType="com.lagou.entity.CourseSection">
            <result property="id" column="cs_id" jdbcType="OTHER"/>
            <result property="courseId" column="cs_course_id"
jdbcType="INTEGER"/>
            <result property="sectionName" column="section_name"
jdbcType="VARCHAR"/>
            <result property="description" column="cs_description"
jdbcType="VARCHAR"/>
```

```
            <result property="createTime" column="cs_create_time"
jdbcType="TIMESTAMP"/>
            <result property="updateTime" column="cs_update_time"
jdbcType="TIMESTAMP"/>
            <result property="isDe" column="is_de" jdbcType="OTHER"/>
            <result property="orderNum" column="cs_order_num"
jdbcType="INTEGER"/>
            <result property="status" column="cs_status" jdbcType="INTEGER"/>

            <!--N：小节-->
            <collection property="courseLessons"
ofType="com.lagou.entity.CourseLesson">
                <result property="id" column="cl_id" jdbcType="OTHER"/>
                <result property="courseId" column="cl_course_id"
jdbcType="INTEGER"/>
                <result property="sectionId" column="cl_section_id"
jdbcType="INTEGER"/>
                <result property="theme" column="theme" jdbcType="VARCHAR"/>
                <result property="duration" column="cl_duration"
jdbcType="INTEGER"/>
                <result property="isFree" column="is_free" jdbcType="OTHER"/>
                <result property="createTime" column="cl_create_time"
jdbcType="TIMESTAMP"/>
                <result property="updateTime" column="cl_update_time"
jdbcType="TIMESTAMP"/>
                <result property="isDel" column="cl_is_del" jdbcType="OTHER"/>
                <result property="orderNum" column="cl_order_num"
jdbcType="INTEGER"/>
                <result property="status" column="cl_status"
jdbcType="INTEGER"/>

                <!-- 1：小节视频-->
                <association property="courseMedia"
javaType="com.lagou.entity.CourseMedia">
                    <result property="id" column="cm_id" jdbcType="INTEGER"/>
                    <result property="courseId" column="cm_course_id"
jdbcType="INTEGER"/>
                    <result property="sectionId" column="cm_section_id"
jdbcType="INTEGER"/>
                    <result property="lessonId" column="cm_lesson_id"
jdbcType="INTEGER"/>
                    <result property="coverImageUrl" column="cover_image_url"
jdbcType="VARCHAR"/>
                    <result property="duration" column="cm_duration"
jdbcType="VARCHAR"/>
                    <result property="fileEdk" column="file_edk"
jdbcType="VARCHAR"/>
                    <result property="fileSize" column="file_size"
jdbcType="OTHER"/>
                    <result property="fileName" column="file_name"
jdbcType="VARCHAR"/>
                    <result property="fileDk" column="file_dk"
jdbcType="VARCHAR"/>
                    <result property="createTime" column="cm_create_time"
jdbcType="TIMESTAMP"/>
                    <result property="updateTime" column="cm_update_time"
jdbcType="TIMESTAMP"/>
```

```
                    <result property="isDel" column="cm_is_del"
jdbcType="OTHER"/>
                    <result property="durationNum" column="duration_num"
jdbcType="INTEGER"/>
                    <result property="fileId" column="file_id"
jdbcType="VARCHAR"/>
                </association>
            </collection>
        </collection>

    </resultMap>

    <sql id="courseInfo">
        select
        c.id
c_id,`course_name`,`brief`,`price`,`price_tag`,`discounts`,`discounts_tag`,`cour
se_description_mark_down`,`course_description`,`course_img_url`,`is_new`,`is_new
_des`,`last_operator_id`,`auto_online_time`,c.create_time
c_create_time,c.update_time c_update_time,c.is_del
c_is_del,`total_duration`,`course_list_img`,c.status
c_status,`sort_num`,`preview_first_field`,`preview_second_field`,`sales`,
        t.id t_id,t.course_id
t_course_id,`teacher_name`,`position`,t.description t_description,t.create_time
t_create_time,t.update_time t_update_time,t.is_del t_is_del,
        cs.id cs_id,cs.course_id cs_course_id,`section_name`,cs.description
cs_description,cs.create_time cs_create_time,cs.update_time
cs_update_time,`is_de`,cs.order_num cs_order_num ,cs.status cs_status,
        cl.id cl_id,cl.course_id cl_course_id,cl.section_id
cl_section_id,`theme`,cl.duration cl_duration,`is_free`,cl.create_time
cl_create_time,cl.update_time cl_update_time,cl.is_del cl_is_del,cl.order_num
cl_order_num,cl.status cl_status,
        cm.id cm_id,cm.course_id cm_course_id,cm.section_id cm_section_id
,cm.lesson_id cm_lesson_id,`cover_image_url`,cm.duration
cm_duration,`file_edk`,`file_size`,`file_name`,`file_dk`,cm.create_time
cm_create_time,cm.update_time cm_update_time,cm.is_del
cm_is_del,`duration_num`,`file_id`,
        ac.id ac_id,ac.course_id
ac_course_id,`begin_time`,`end_time`,`amount`,`stock`,ac.status
ac_status,ac.is_del ac_is_del,`remark`,ac.create_time
ac_create_time,`create_user`,ac.update_time ac_update_time,`update_user`
    from activity_course ac right join course c on  c.id = ac.course_id
        inner join teacher t on c.id = t.course_id
        inner join course_section cs on c.id = cs.course_id
        inner join course_lesson cl on cl.section_id = cs.id
        left join course_media cm on cm.lesson_id = cl.id
    </sql>

    <select id="getAllCourse" resultMap="CourseMap">
        <include refid="courseInfo"/>
        order by amount desc , c_id , ac_create_time desc
    </select>
</mapper>
```

```
package mapper;

import com.lagou.entity.Course;
import org.springframework.stereotype.Service;
```

```
import java.util.List;

@Service
public interface CourseDao {

    /**
     * 查询全部课程信息
     * @return
     */
    List<Course> getAllCourse();
}
```

```
@RunWith(SpringJUnit4ClassRunner.class)
@ContextConfiguration(locations = { "classpath:spring/spring-dao.xml" })
public class TestCourse {
    @Autowired
    private CourseDao courseDao;

    @Test
    public void getAllCourse(){
        List<Course> list = courseDao.getAllCourse();
        for(Course course : list){
            //if( course.getId().toString() .equals("8") ){ //查询第8门课
                String flag = course.getActivityCourse()!=null ? "【火爆活动
中】":"";
                System.out.println("课程: "+flag + course.getId() +"->"+
course.getCourseName());

                for(CourseSection cs : course.getCourseSections() ){
                    System.out.println("\t\t章节" + cs.getId() +"--->" +
cs.getSectionName());
                    for(CourseLesson cl : cs.getCourseLessons()){
                        if(cl.getCourseMedia() != null){
                            System.out.println("\t\t\t小节" + cl.getId() +"--->"
+ cl.getTheme()+",视频: " + cl.getCourseMedia().getFileId()+",时长: 【"+
cl.getCourseMedia().getDuration()+"】");
                        }else{
                            System.out.println("\t\t\t小节" + cl.getId() +"--->"
+ cl.getTheme()+",视频: 【待上传】,时长: 【00:00】");
                        }
                    }
                }
            //}
        }
    }
}
```

- lagou-edu-service

```java
package com.lagou.course;

import com.lagou.entity.Course;
import java.util.List;

public interface CourseService {
    /**
     * 查询全部课程信息
     * @return
     */
    List<Course> getAllCourse();
}
```

```java
package com.lagou.course.impl;

import com.alibaba.dubbo.config.annotation.Service;
import com.lagou.course.CourseService;
import com.lagou.entity.Course;
import mapper.CourseDao;
import org.springframework.beans.factory.annotation.Autowired;
import java.util.List;

@Service
public class CourseServiceImpl implements CourseService {

    @Autowired
    private CourseDao courseDao;

    public List<Course> getAllCourse() {
        return courseDao.getAllCourse();
    }
}
```

- lagou-edu-web

```java
package com.lagou.course;

import com.lagou.entity.Course;
import java.util.List;

public interface CourseService {
    /**
     * 查询全部课程信息
     * @return
     */
    List<Course> getAllCourse();
}
```

```java
package com.lagou.course.controller;

import com.alibaba.dubbo.config.annotation.Reference;
import com.lagou.course.CourseService;
import com.lagou.entity.Course;
```

```
import com.lagou.entity.User;
import com.lagou.entity.UserDTO;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;
import java.util.List;

@RestController
@RequestMapping("course")
public class CourseController {

    @Reference // 远程消费
    private CourseService courseService;

    @GetMapping("getAllCourse")
    public List<Course> getAllCourse() {
        List<Course> list = courseService.getAllCourse();
        return list;
    }
}
```

2.2.2 已购课程

功能名称	已购
功能描述	1、用户未登录则显示为空。 2、用户登录，通过用户ID调用服务获取已购课程
功能接口	/course/getCourseByUserId/{userid}

- lagou-edu-entity（同上）
- lagou-edu-dao

```
<select id="getCourseByUserId" resultMap="CourseMap">
    <include refid="courseInfo"/>
    where  c.id in( select course_id from user_course_order where status = 20
and  user_id = #{userId} )
    order by amount desc , c_id , ac_create_time desc
</select>
```

```
@Service
public interface CourseDao {

    /**
     * 查询已登录用户购买的全部课程信息
     * @return
     */
    List<Course> getCourseByUserId(@Param("userId") String userId);
}
```

- lagou-edu-service

```
public interface CourseService {
    /**
     * 查询已登录用户购买的全部课程信息
     * @return
     */
    List<Course> getCourseByUserId(String userId);
}
```

```
@Service
public class CourseServiceImpl implements CourseService {

    @Autowired
    private CourseDao courseDao;

    public List<Course> getCourseByUserId(String userId) {
        return courseDao.getCourseByUserId(userId);
    }
}
```

- lagou-edu-web

```
public interface CourseService {
    /**
     * 查询已登录用户购买的全部课程信息
     * @return
     */
    List<Course> getCourseByUserId(String userId);
}
```

```
@RestController
@RequestMapping("course")
public class CourseController {

    @Reference // 远程消费
    private CourseService courseService;

    @GetMapping("getCourseByUserId/{userid}")
    public List<Course> getCourseByUserId( @PathVariable("userid") String userid
) {
        List<Course> list = courseService.getCourseByUserId(userid);
        return list;
    }
}
```

2.2.3 课程详情

功能名称	课程详情
功能描述	1、用户如果没有登录， 通过课程ID获取课程的详细信息中会显示“立即购买”按钮。 点击够买按钮会显示登录。同时会展示课程目录，课程信息和老师信息。 注意课程信息是展示 字段course_description中的信息。 2、用户登录后，我们通过课程Id获取课程信息后， 会通过用户ID和课程ID调用订单服务来获取来判断用户是否购买， 如果够买则显示“立即学习”，否则显示“立即够买”， 同时会展示课程目录，课程信息和老师信息。 注意课程信息是展示 字段course_description中的信息。
功能接口	/course/getCourseById/{courseid}

- lagou-edu-entity （同上）
- lagou-edu-dao

```
<select id="getCourseById" resultMap="CourseMap">
    <include refid="courseInfo"/>
    where c.id = #{courseid}
</select>
```

```
@Service
public interface CourseDao {
    /**
     * 查询某门课程的详细信息
     * @param courseid 课程编号
     * @return
     */
    Course getCourseById(@Param("courseid") Integer courseid);
}
```

```
@RunWith(SpringJUnit4ClassRunner.class)
@ContextConfiguration(locations = {"classpath:spring/spring-dao.xml"})
public class TestCourse {
    @Autowired
    private CourseDao courseDao;

    @Test
    public void getCourseById() {
        Course course = courseDao.getCourseById(7);
        String flag = course.getActivityCourse() != null ? "【火爆活动中】" : "";
        System.out.println("课程：" + flag + course.getId() + "->" +
course.getCourseName());

        for (CourseSection cs : course.getCourseSections()) {
            System.out.println("\t\t章节" + cs.getId() + "--->" +
cs.getSectionName());
            for (CourseLesson cl : cs.getCourseLessons()) {
                if (cl.getCourseMedia() != null) {
                    System.out.println("\t\t\t小节" + cl.getId() + "--->" +
cl.getTheme() + ",视频：" + cl.getCourseMedia().getFileId() + ",时长：【" +
cl.getCourseMedia().getDuration() + "】");
                } else {
```

```
                    System.out.println("\t\t\t小节" + cl.getId() + "--->" +
cl.getTheme() + ",视频：【待上传】,时长：【00:00】");
                }
            }
        }
    }
}
```

- lagou-edu-service

```
public interface CourseService {
    Course getCourseById(Integer courseid);
}
```

```
@Service // 暴露服务
public class CourseServiceImpl implements CourseService {

    public Course getCourseById(Integer courseid) {
        return courseDao.getCourseById(courseid);
    }
}
```

- lagou-edu-web

```
public interface CourseService {
    /**
     * 查询某门课程的详细信息
     * @param courseid 课程编号
     * @return
     */
    Course getCourseById(Integer courseid);
}
```

```
@RestController
@RequestMapping("course")
public class CourseController {

    @Reference // 远程消费
    private CourseService courseService;

    @GetMapping("getCourseById/{courseid}")
    public Course getCourseById(@PathVariable("courseid")Integer courseid) {
        Course course = courseService.getCourseById(courseid);
        return course;
    }
}
```

2.2.4 课程播放

功能名称	课程播放
功能描述	前端通过课时ID调用接口 1 获取课程阿里的fileId，然后通过fileId调用接口 2 获取视频播放的playAuth,前端会通过playAuth调用阿里的播放器， 阿里会回调我们的接口 3 ，我们会通过阿里的回调来判断是否是合法用户，用户是否购买课程， 这样可以防止我们的播放链接被盗后也不能被播放，确保我们课程版权的安全。
功能接口	1、通过课时ID获取媒体信息 /front/course/media/getByLessonId 2、获取阿里播放key /front/course/media/alikey 3、获取阿里播放key (用于阿里的回调) /front/course/media/alikey

2.3 订单模块

2.3.1 购买/生成订单

功能名称	下单
功能描述	用户选好课程点击立即购买,调用后端接口,开始创建商品订单。 需要校验必要数据：用户是否登录， 接口的请求入参，课程是否存在，课程是否上架， 该用户之前是否成功购买过该课程， 如校验不通过提示相应的文案。 查看该用户是否存在有效未成功的订单， 如果存在直接返回对应的订单号， 如果不存在创建新的课程订单，返回新的订单号。
功能接口	/order/saveOrder/{userid}/{courseid}/{acid}/{stype}

- lagou-edu-dao

```
<insert id="saveOrder">
insert  into `user_course_order`
(`order_no`,`user_id`,`course_id`,`activity_course_id`,`source_type`,`status`,`c
reate_time`,`update_time`,`is_del`)
values
(#{orderNo},#{user_id},#{course_id},#{activity_course_id},#
{source_type},0,sysdate(),sysdate(),0)
</insert>
```

```
@Service
public interface OrderDao {

    /**
     * 生成订单
     * @param orderNo 订单编号
     * @param user_id 用户编号
     * @param course_id 课程编号
     * @param activity_course_id 活动课程编号
     * @param source_type 订单来源类型
     */
    void saveOrder(@Param("orderNo")String orderNo,@Param("user_id")String
user_id,@Param("course_id")String course_id,@Param("activity_course_id")String
activity_course_id,@Param("source_type")String source_type);
}
```

- lagou-edu-service

```
public interface OrderService {

    /**
     * 生成订单
     * @param orderNo 订单编号
     * @param user_id 用户编号
     * @param course_id 课程编号
     * @param activity_course_id 活动课程编号
     * @param source_type 订单来源类型
     */
    void saveOrder(String orderNo, String user_id, String course_id,String
activity_course_id, String source_type);
}
```

```
@Service //暴露服务
public class OrderServiceImpl implements OrderService {
    @Autowired
    private OrderDao orderDao;
    public void saveOrder(String orderNo, String user_id, String course_id,
String activity_course_id, String source_type) {
        orderDao.saveOrder(orderNo, user_id, course_id, activity_course_id,
source_type);
    }
}
```

```
@RunWith(SpringJUnit4ClassRunner.class)
@ContextConfiguration(locations = { "classpath*:spring/spring-*.xml" })
public class TestOrder {
    @Autowired
    private OrderService orderService;

    @Test
    public void saveOrder(){
        String orderNo= UUID.randomUUID().toString();
        String user_id = "100030011";
```

```
        String course_id = "7";
        String activity_course_id = "0"; // 0表示，本课程没有活动
        String source_type = "1";

 orderService.saveOrder(orderNo,user_id,course_id,activity_course_id,source_type
);
    }
}
```

- lagou-edu-web

```
public interface OrderService {
    /**
     * 生成订单
     * @param orderNo 订单编号
     * @param user_id 用户编号
     * @param course_id 课程编号
     * @param activity_course_id 活动课程编号
     * @param source_type 订单来源类型
     */
    void saveOrder(String orderNo, String user_id, String course_id,String
activity_course_id, String source_type);
}
```

```
@RestController
@RequestMapping("order")
public class OrderController {

    @Reference // 远程消费
    private OrderService orderService;

    @GetMapping("saveOrder/{userid}/{courseid}/{acid}/{stype}")
    public String saveOrder(@PathVariable("userid")String user_id ,
@PathVariable("courseid")String course_id,@PathVariable("acid")String
activity_course_id,@PathVariable("stype")String source_type) {
        String orderNo = UUID.randomUUID().toString();
        orderService.saveOrder(orderNo, user_id, course_id, activity_course_id,
source_type);
        return orderNo;
    }
}
```

2.3.2 订单操作

功能名称	下单
功能描述	1、修改订单状态。 2、删除订单。 3、查询已登录用户的全部订单。
功能接口	/order/updateOrder/{orderNo} /order/deleteOrder/{orderNo} /order/getOrdersByUserId/{userid}

- lagou-edu-dao

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
"http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="mapper.OrderDao">

    <resultMap type="com.lagou.entity.UserCourseOrder" id="orderMap">
        <result property="id" column="id" jdbcType="INTEGER"/>
        <result property="orderNo" column="order_no" jdbcType="VARCHAR"/>
        <result property="userId" column="user_id" jdbcType="OTHER"/>
        <result property="courseId" column="course_id" jdbcType="OTHER"/>
        <result property="activityCourseId" column="activity_course_id"
jdbcType="INTEGER"/>
        <result property="sourceType" column="source_type" jdbcType="OTHER"/>
        <result property="status" column="status" jdbcType="OTHER"/>
        <result property="createTime" column="create_time"
jdbcType="TIMESTAMP"/>
        <result property="updateTime" column="update_time"
jdbcType="TIMESTAMP"/>
        <result property="isDel" column="is_del" jdbcType="OTHER"/>
    </resultMap>

    <update id="updateOrder">
        update user_course_order set status = #{status} where order_no = #
{orderNo} and is_del = 0
    </update>

    <update id="deleteOrder">
        update user_course_order set is_del= 1 where order_no = #{orderNo}
    </update>

    <select id="getOrdersByUserId" resultMap="orderMap">
        select * from user_course_order where is_del = 0 and user_id = #{userId}
    </select>

</mapper>
```

```
@Service
public interface OrderDao {

    /**
     * 修改订单状态
```

```
     * @param orderNo 订单编号
     * @param status 订单状态 0已创建 10已支付 20已完成 30已取消 40已过期
     * @return 受影响的行数
     */
    Integer updateOrder( @Param("orderNo") String orderNo,@Param("status") int
status );

    /**
     * 删除订单
     * @param orderNo 订单编号
     * @return 受影响的行数
     */
    Integer deleteOrder(@Param("orderNo") String orderNo);

    /**
     * 查询登录用户的全部订单
     * @param userId 用户编号
     * @return 所有订单
     */
    List<UserCourseOrder> getOrdersByUserId(@Param("userId") String userId);

}
```

- lagou-edu-service

```
public interface OrderService {

    /**
     * 修改订单状态
     * @param orderNo 订单编号
     * @param status 订单状态 0已创建 10已支付 20已完成 30已取消 40已过期
     * @return 受影响的行数
     */
    Integer updateOrder( String orderNo,int status );

    /**
     * 删除订单
     * @param orderNo 订单编号
     * @return 受影响的行数
     */
    Integer deleteOrder(String orderNo);

    /**
     * 查询登录用户的全部订单
     * @param userId 用户编号
     * @return 所有订单
     */
    List<UserCourseOrder> getOrdersByUserId(String userId);
}
```

```
@Service //暴露服务
public class OrderServiceImpl implements OrderService {
    @Autowired
    private OrderDao orderDao;
```

```
    public Integer updateOrder(String orderNo, int status) {
        return orderDao.updateOrder(orderNo,status);
    }

    public Integer deleteOrder(String orderNo) {
        return orderDao.deleteOrder(orderNo);
    }

    public List<UserCourseOrder> getOrdersByUserId(String userId) {
        return orderDao.getOrdersByUserId(userId);
    }
}
```

- lagou-edu-web

```
public interface OrderService {
    /**
     * 修改订单状态
     * @param orderNo 订单编号
     * @param status 订单状态 0已创建 10已支付 20已完成 30已取消 40已过期
     * @return 受影响的行数
     */
    Integer updateOrder( String orderNo,int status );

    /**
     * 删除订单
     * @param orderNo 订单编号
     * @return 受影响的行数
     */
    Integer deleteOrder(@Param("orderNo") String orderNo);

    /**
     * 查询登录用户的全部订单
     * @param userId 用户编号
     * @return 所有订单
     */
    List<UserCourseOrder> getOrdersByUserId(String userId);
}
```

```
@RestController
@RequestMapping("order")
public class OrderController {

    @Reference // 远程消费
    private OrderService orderService;

    @GetMapping("updateOrder/{orderno}/{status}")
    public Integer updateOrder(@PathVariable("orderno")String orderno ,
@PathVariable("status")Integer status) {
        Integer integer = orderService.updateOrder(orderno, status);
        return integer;
    }

    @GetMapping("deleteOrder/{orderno}")
    public Integer deleteOrder(@PathVariable("orderno")String orderno ) {
```

```
        Integer integer = orderService.deleteOrder(orderno);
        return integer;
    }

    @GetMapping("getOrdersByUserId/{userid}")
    public List<UserCourseOrder> getOrdersByUserId(@PathVariable("userid")String
userid ) {
        List<UserCourseOrder> list = orderService.getOrdersByUserId(userid);
        return list;
    }
}
```

2.4 留言模块

2.4.1 保存留言

功能名称	保存课程留言
功能描述	1. 课程Id和评论内容都不能为空 2. 保存留言时需要保存用户的ID 以便于查看留言时候判断留言是否是某个用户写的。
功能接口	/course/comment/saveCourseComment

- lagou-edu-entity

```
@Data
@AllArgsConstructor
@NoArgsConstructor
@ToString
public class CourseComment implements Serializable {
    private static final long serialVersionUID = 922554392538715061L;
```

- lagou-edu-dao

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
"http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="mapper.CourseCommentDao">

    <resultMap type="com.lagou.entity.CourseComment" id="commentMap">
        <result property="id" column="id" jdbcType="OTHER"/>
        <result property="courseId" column="course_id" jdbcType="INTEGER"/>
        <result property="sectionId" column="section_id" jdbcType="INTEGER"/>
        <result property="lessonId" column="lesson_id" jdbcType="INTEGER"/>
        <result property="userId" column="user_id" jdbcType="INTEGER"/>
        <result property="userName" column="user_name" jdbcType="VARCHAR"/>
        <result property="parentId" column="parent_id" jdbcType="INTEGER"/>
        <result property="isTop" column="is_top" jdbcType="OTHER"/>
```

```
        <result property="comment" column="comment" jdbcType="VARCHAR"/>
        <result property="likeCount" column="like_count" jdbcType="INTEGER"/>
        <result property="isReply" column="is_reply" jdbcType="OTHER"/>
        <result property="type" column="type" jdbcType="INTEGER"/>
        <result property="status" column="status" jdbcType="INTEGER"/>
        <result property="createTime" column="create_time"
jdbcType="TIMESTAMP"/>
        <result property="updateTime" column="update_time"
jdbcType="TIMESTAMP"/>
        <result property="isDel" column="is_del" jdbcType="OTHER"/>
        <result property="lastOperator" column="last_operator"
jdbcType="INTEGER"/>
        <result property="isNotify" column="is_notify" jdbcType="OTHER"/>
        <result property="markBelong" column="mark_belong" jdbcType="OTHER"/>
        <result property="replied" column="replied" jdbcType="OTHER"/>
    </resultMap>

   <insert id="saveComment">
       insert  into
`course_comment`(`course_id`,`section_id`,`lesson_id`,`user_id`,`user_name`,`par
ent_id`,`is_top`,`comment`,`like_count`,`is_reply`,`type`,`status`,`create_time`
,`update_time`,`is_del`,`last_operator`,`is_notify`,`mark_belong`,`replied`)
       values
       (#{courseId},#{sectionId},#{lessonId},#{userId},#{userName},#
{parentId},0,#{comment},0,0,#{type},0,sysdate(),sysdate(),0,#
{lastOperator},1,0,0)
   </insert>

</mapper>
```

```
@Service
public interface CourseCommentDao {
    /**
     * 保存留言
     * @param comment 留言内容对象
     * @return 受影响的行数
     */
    Integer saveComment(CourseComment comment);
}
```

- lagou-edu-service

```
public interface CommentService {
    /**
     * 保存留言
     * @param comment 留言内容对象
     * @return 受影响的行数
     */
    Integer saveComment(CourseComment comment);
}
```

```
@Service // 暴露服务
public class CommentServiceImpl implements CommentService {

    @Autowired
    private CourseCommentDao courseCommentDao;

    public Integer saveComment(CourseComment comment) {
        return courseCommentDao.saveComment(comment);
    }
}
```

- lagou-edu-web

```
public interface CommentService {
    /**
     * 保存留言
     * @param comment 留言内容对象
     * @return 受影响的行数
     */
    Integer saveComment(CourseComment comment);
}
```

```
@RestController
@RequestMapping("course")
public class CommentController {

    @Reference // 远程消费
    private CommentService commentService;


    @GetMapping("comment/saveCourseComment")
    public Object saveCourseComment(){
        CourseComment comment = new CourseComment();
        comment.setCourseId(7); // 课程编号
        comment.setSectionId(8); // 章节编号
        comment.setLessonId(10);// 小节编号
        comment.setUserId(100030011); // 用户编号
        comment.setUserName("Angier"); // 用户昵称
        comment.setParentId(0); //没有父id
        comment.setComment("old tie, 666!");// 留言内容
        comment.setType(0); // 0用户留言
        comment.setLastOperator(100030011); //最后操作的用户编号
        Integer i = commentService.saveComment(comment);
        System.out.println(i);
        return null;
    }
}
```

2.4.2 留言列表

某门课程的全部留言

功能名称	获取课程留言
功能描述	1. 通过课程Id、页号和用户ID分页获取留言信息 每页默认条数是20条。 当用户Id不为空的时候，前端标识出此用户点赞的留言。 2. 留言中返回重要内容中包括： 评论内容、本人是否点赞、用户昵称、是否点过赞标识和点赞数量。 3. 留言按照点赞数量和创建时间的倒序显示。
功能接口	/course/comment/getCourseCommentList/{courseid}/{pageIndex}/{pageSize}

- lagou-edu-entity (同上)
- lagou-edu-dao

```
<select id="getCommentsByCourseId" resultMap="commentMap">
    select * from course_comment
    where is_del = 0
    and course_id = #{courseid}
    order by is_top desc , like_count desc , create_time desc
    limit #{offset}, #{pageSize}
</select>
```

```
@Service
public interface CourseCommentDao {

    /**
     * 某个课程的全部留言（分页）
     * @param courseid 课程编号
     * @param offset 数据偏移
     * @param pageSize 每页条目数
     * @return 留言集合
     */
    List<CourseComment> getCommentsByCourseId(@Param("courseid")Integer
courseid,
                                              @Param("offset")Integer offset,
                                              @Param("pageSize")Integer
pageSize);
}
```

- lagou-edu-service

```
public interface CommentService {

    /**
     * 某个课程的全部留言（分页）
     * @param courseid 课程编号
     * @param offset 数据偏移
     * @param pageSize 每页条目数
     * @return 留言集合
     */
    List<CourseComment> getCommentsByCourseId(@Param("courseid")Integer
courseid,
                                              @Param("offset")Integer offset,
                                              @Param("pageSize")Integer
pageSize);
}
```

```
@Service // 暴露服务
public class CommentServiceImpl implements CommentService {

    @Autowired
    private CourseCommentDao courseCommentDao;

    public List<CourseComment> getCommentsByCourseId(Integer courseid, Integer
offset, Integer pageSize) {
        return courseCommentDao.getCommentsByCourseId(courseid, offset,
pageSize);
    }
}
```

```
@RunWith(SpringJUnit4ClassRunner.class)
@ContextConfiguration(locations = { "classpath*:spring/spring-*.xml" })
public class TestComment {

    @Autowired
    private CommentService commentService;

    @Test
    public void getCommentsByCourseId(){
        int pageSize = 20;
        int pageIndex = 3; //页码（第几页）

        List<CourseComment> list = commentService.getCommentsByCourseId(1,
(pageIndex-1)*pageSize, pageSize);
        for(int i = 0;i<list.size();i++){
            CourseComment comment = list.get(i);
            System.out.println((i+1)+"、楼 【"+comment.getUserName()+"】 说： " +
comment.getComment());
        }
    }
}
```

- lagou-edu-web

```
public interface CommentService {

    /**
     * 某个课程的全部留言（分页）
     * @param courseid 课程编号
     * @param offset 数据偏移
     * @param pageSize 每页条目数
     * @return 留言集合
     */
    List<CourseComment> getCommentsByCourseId(@Param("courseid")Integer
courseid,
                                              @Param("offset")Integer offset,
                                              @Param("pageSize")Integer
pageSize);
}
```

```
@RestController
@RequestMapping("course")
public class CommentController {

    @Reference // 远程消费
    private CommentService commentService;


 @GetMapping("comment/getCourseCommentList/{courseid}/{pageIndex}/{pageSize}")
    public List<CourseComment> getCommentsByCourseId(@PathVariable("courseid")
Integer courseid,
                                                     @PathVariable("pageIndex")
Integer pageIndex,
                                                     @PathVariable("pageSize")
Integer pageSize){
        List<CourseComment> list =
commentService.getCommentsByCourseId(courseid, pageIndex, pageSize);
        return list;
    }
}
```

2.4.3 留言 点赞/取消赞

功能名称	点赞
功能描述	点赞 1. 先根据用户ID和留言ID查看留言是否存在点赞信息 如果存在则更新删除状态，否则保存点赞信息。 2. 保存点赞后，需要更新对应留言的点赞数量。 取消赞 1.通过用户ID和留言ID获取用户的点赞信息 2.然后将记录进行逻辑删除，然后更新对应记录的点赞数量

功能接口	/course/comment/saveFavorite/{commentid}/{userid} // 点赞
	
	/course/comment/cancelFavorite/{commentid}/{userid} // 取消赞

- lagou-edu-entity (无)
- lagou-edu-dao

```
<!--查看某个用户的某条留言是否点过赞-->
<select id="existsFavorite" resultType="Integer">
    select count(*) from course_comment_favorite_record where comment_id = #
{cid} and user_id = #{uid}
</select>

<!--没有点过赞，保存点赞信息-->
<insert id="saveCommentFavorite">
    insert  into
`course_comment_favorite_record`(user_id,`comment_id`,`is_del`,`create_time`,`up
date_time`) values
(#{uid},#{cid},0,sysdate(),sysdate())
</insert>

<!--修改点赞的状态，0表示点赞,1表示取消赞-->
<update id="updateFavoriteStatus">
    update course_comment_favorite_record set is_del = #{status} where
comment_id = #{cid} and user_id = #{uid}
</update>

<!--点赞之后，赞的数量+1；取消赞之后，赞的数量-1-->
<update id="updateLikeCount">
    update course_comment set like_count = like_count + #{x} where id = #
{comment_id}
</update>
```

```
@Service
public interface CourseCommentDao {
    /**
     * 查看某个用户的某条留言是否点过赞
     * @param comment_id 留言编号
     * @param userid 用户编号
     * @return 0：没点过赞，1：点过赞
     */
    Integer existsFavorite(@Param("cid")Integer comment_id,@Param("uid")Integer
userid);
```

```
    /**
     * 保存点赞信息
     * @param comment_id 留言编号
     * @param userid 用户编号
     * @return 0：保存失败，1：保存成功
     */
    Integer saveCommentFavorite(@Param("cid")Integer
comment_id,@Param("uid")Integer userid);

    /**
     * 更新点赞信息的状态（将is_del=0，表示已赞）
     * @param status 状态，0：已赞，1：取消赞
     * @param comment_id 留言编号
     * @param userid 用户编号
     * @return 0：保存失败，1：保存成功
     */
    Integer updateFavoriteStatus( @Param("status")Integer status,
@Param("cid")Integer comment_id,@Param("uid")Integer userid);

    /**
     * 更新点赞的数量
     * @param x +1的话，赞的数量增加，-1的话，赞的数量减少
     * @param comment_id 某条留言的编号
     * @return 0：保存失败，1：保存成功
     */
    Integer updateLikeCount(@Param("x")Integer x ,@Param("comment_id")Integer
comment_id);
}
```

- lagou-edu-service

```
public interface CommentService {
    /**
     * 点赞
     * @param comment_id 留言编号
     * @param userid 用户编号
     * @return 0：保存失败，1：保存成功
     */
    Integer saveFavorite(Integer comment_id,Integer userid);

    /**
     * 取消赞
     * @param comment_id 留言编号
     * @param userid 用户编号
     * @return 0：保存失败，1：保存成功
     */
    Integer cancelFavorite(Integer comment_id,Integer userid);
}
```

```
@Service // 暴露服务
public class CommentServiceImpl implements CommentService {

    @Autowired
    private CourseCommentDao courseCommentDao;
```

```
    /**
     *点赞：
     * 先查看当前用户对这条留言是否点过赞，
     * 如果点过：修改is_del状态即可，取消赞
     * 如果没点过：保存一条点赞的信息
     *
     * 最终，更新赞的数量
      */
    public Integer saveFavorite(Integer comment_id, Integer userid) {
        Integer i = courseCommentDao.existsFavorite(comment_id, userid);
        int i1 = 0;
        int i2 = 0;
        if(i == 0){ //没点过赞
            i1 = courseCommentDao.saveCommentFavorite(comment_id,userid);
        }else{
            i1 = courseCommentDao.updateFavoriteStatus(0,comment_id,userid);
        }
        i2 = courseCommentDao.updateLikeCount(1,comment_id);

        if(i1==0 || i2==0){
            throw  new RuntimeException("点赞失败！");
        }
        return comment_id;
    }

    /**
     * 删除点赞的信息（is_del = 1）
     * 更新留言赞的数量-1
     * @param comment_id 留言编号
     * @param userid 用户编号
     * @return 0:失败，1：成功
     */
    public Integer cancelFavorite(Integer comment_id, Integer userid) {
        Integer i1 = courseCommentDao.updateFavoriteStatus(1, comment_id,
userid);
        Integer i2 = courseCommentDao.updateLikeCount(-1,comment_id);

        if(i1==0 || i2==0){
            throw  new RuntimeException("取消赞失败！");
        }
        return i2;
    }
}
```

- lagou-edu-web

```
public interface CommentService {
    /**
     * 点赞
     * @param comment_id 留言编号
     * @param userid 用户编号
     * @return 0：保存失败，1：保存成功
     */
    Integer saveFavorite(Integer comment_id,Integer userid);
```

```
    /**
     * 取消赞
     * @param comment_id 留言编号
     * @param userid 用户编号
     * @return 0：保存失败，1：保存成功
     */
    Integer cancelFavorite(Integer comment_id,Integer userid);
}
```

```
@RestController
@RequestMapping("course")
public class CommentController {

    @Reference // 远程消费
    private CommentService commentService;

    // 点赞
    @GetMapping("comment/saveFavorite/{commentid}/{userid}")
    public Integer saveFavorite(@PathVariable("commentid") Integer
commentid,@PathVariable("userid") Integer userid){
        Integer integer = commentService.saveFavorite(commentid, userid);
        return integer;
    }

    // 取消赞
    @GetMapping("comment/cancelFavorite/{commentid}/{userid}")
    public Integer cancelFavorite(@PathVariable("commentid") Integer
commentid,@PathVariable("userid") Integer userid){
        Integer integer = commentService.cancelFavorite(commentid, userid);
        return integer;
    }
}
```