

任务二 课程管理模块开发1

1. 开发流程

1.1 需求分析



1.2 数据库表分析

这里展示的是我们需要使用的部分表字段

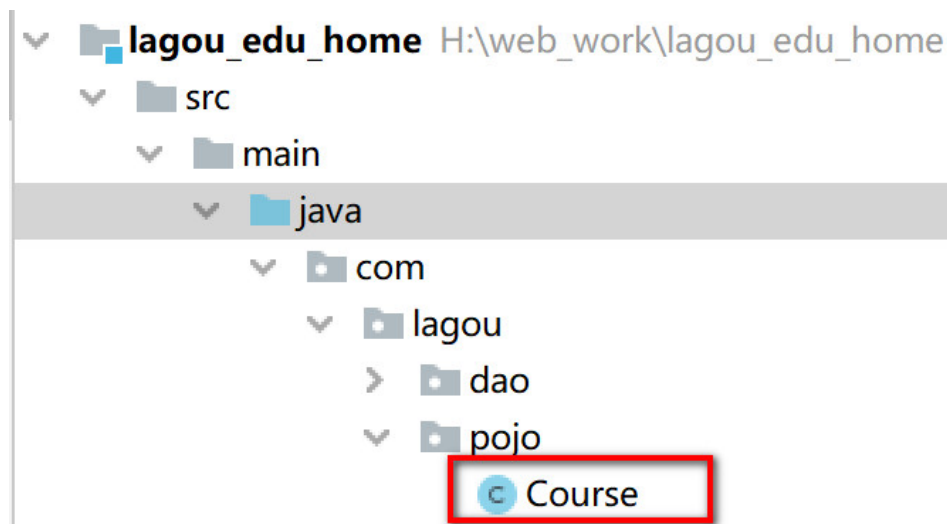
course 课程表

id	课程id
course_name	课程名称
brief	课程简介
teacher_name	讲师名称
teacher_info	讲师介绍
preview_first_field	课程概述
price	课程原价
price_tag	原价标签
discounts	课程优惠价
course_img_url	分享图片url

share_title	分享标题
share_description	分享描述
course_description	课程描述
sort_num	排序
status	课程状态,0-草稿,1-上架
create_time	创建时间
update_time	修改时间
is_del	是否删除 0-未删除 1-已删除

1.3 实体类设计

根据数据库中的Course表,对应创建 Course.java



1. 使用 @JSONField(ordinal = int类型的值), 指定排序的值,生成JSON时会按照指定顺序进行排序
2. 使用 @JSONField(serialize = false),排除不需要转换的字段,另外fastjson还会自动排除为空的字段

```
/**
 * 课程类
 * */
@Data
public class Course implements Serializable {

    //使用 JSONField 设置ordinal的值,来对转换成的JSON数据进行排序
```

```
//课程ID
@JSONField(ordinal = 1)
private int id;

//课程名称
@JSONField(ordinal = 2)
private String course_name;

//课程介绍
@JSONField(ordinal = 3)
private String brief;

//讲师名称
@JSONField(ordinal = 4)
private String teacher_name;

//讲师介绍
@JSONField(ordinal = 5)
private String teacher_info;

//课程原价
@JSONField(ordinal = 6)
private double price;

//原价标签
@JSONField(ordinal = 7)
private String price_tag;

//课程优惠价
@JSONField(ordinal = 8)
private double discounts;

//课程概述
@JSONField(ordinal = 9)
private String preview_first_field;

//课程概述第二个字段
@JSONField(ordinal = 10)
private String preview_second_field;

//分享图片url
@JSONField(ordinal = 11)
private String course_img_url;

//分享标题
@JSONField(ordinal = 12)
private String share_title;

//分享描述
@JSONField(ordinal = 13)
private String share_description;

//课程描述
@JSONField(ordinal = 14)
private String course_description;

//排序
@JSONField(ordinal = 15)
```

```
private int sort_num;

//课程状态,0-草稿,1-上架
@JSONField(ordinal = 16)
private int status;

//创建时间
@JSONField(ordinal = 17)
private String create_time;

//修改时间
@JSONField(ordinal = 18)
private String update_time;

//是否删除
@JSONField(ordinal = 19)
private int isDel;

@JSONField(ordinal = 20)
private String share_image_title; //分享图title

//使用JSONField(serialize = false)排除不需要转换的字段

@JSONField(serialize = false)
private int total_course_time; //课时数

@JSONField(serialize = false)
private int sales; //显示销量

@JSONField(serialize = false)
private int actual_sales; //真实销量

@JSONField(serialize = false)
private int is_new; //是否新品

@JSONField(serialize = false)
private String is_new_des; //广告语

@JSONField(serialize = false)
private int last_operator_id; //最后操作者

@JSONField(serialize = false)
private int total_duration; //总时长

@JSONField(serialize = false)
private long course_type; //课程类型

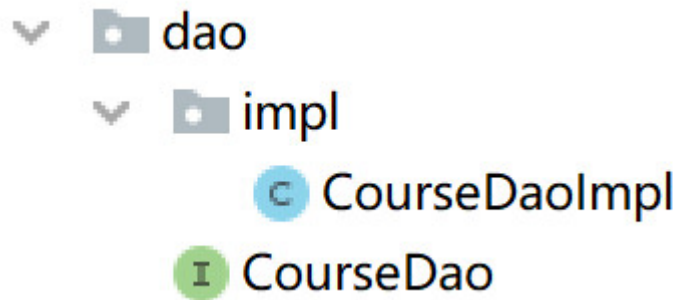
@JSONField(serialize = false)
private String last_notice_time; //最后课程最近通知时间

@JSONField(serialize = false)
private long is_gray; //是否是灰度课程

@JSONField(serialize = false)
private long grade; //级别
```

```
}
```

1.4 Dao接口及实现类编写



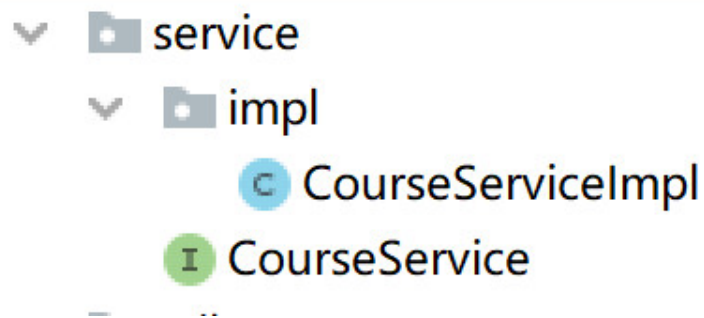
```
/**
 * 课程模块 DAO层接口
 * */
public interface CourseDao {

}
```

```
/**
 * 课程模块 DAO层实现类
 * */
public class CourseDaoImpl implements CourseDao {

}
```

1.5 Service接口及实现类编写



```
/**
 * 课程模块 Service层 接口
 * */
public interface CourseService {

}
```

```
/**
 * 课程模块Service层 实现类
 */
public class CourseServiceImpl implements CourseService {

}
```

1.6 CourseServlet编写

CourseServlet 要继承通用的BaseServlet.

```
@@WebServlet(name="courseServlet",value="/course")
public class CourseServlet extends BaseServlet {

}
```

2. 功能一: 查询课程列表信息

2.1 需求分析

页面分析,需要展示哪些数据

教育/课程管理

+ 新建课程

课程名称

全部

状态

全部

查询

需要展示的数据

ID	课程名称	价格	排序	状态	操作		
3	Flutter实战与进阶	1/98	0	下架	上架	营销信息	配置课时
2	Python编程：Web开发篇	1/98	1	上架	下架	营销信息	配置课时

2.2 编写代码

2.2.1 Dao层编写

1. 修改CourseDao,添加 **findCourseList** 方法

```
接口 CourseDao
//查询课程列表信息
public List<Course> findCourseList();

实现类 CourseDaoImpl
@Override
public List<Course> findCourseList() {

    try {
        //1.创建QueryRunner
        QueryRunner qr = new QueryRunner(DruidUtils.getDataSource());
```

```

        //2.编写SQL
        String sql = "SELECT id,course_name,price,sort_num,STATUS FROM course
        where id_del = ?";

        //3.执行查询
        List<Course> courseList = qr.query(sql, new BeanListHandler<Course>
        (Course.class), 0);

        return courseList;

    } catch (SQLException e) {
        e.printStackTrace();
        return null;
    }
}

```

逻辑删除

- 逻辑删除的本质是**修改操作**，所谓的逻辑删除其实并不是真正的删除，而是在表中将对应的是否删除标识做修改操作。比如: 0是未删除，1是删除。在逻辑上数据是被删除的，但数据本身依然存在库中。

物理删除

- 物理删除就是真正的从数据库中做删除操作了。

2.2.2 Service层编写

修改CourseService 添加 **findCourseList** 方法

```

接口 CourseService
    public List<Course> findCourseList();

实现类 CourseServiceImpl
    //创建 CourseDao
    CourseDao courseDao = new CourseDaoImpl();

    @Override
    public List<Course> findCourseList() {

        //调用Dao 进行查询
        return courseDao.findCourseList();
    }

```

2.2.3 Servlet编写

2.2.3.1 接口开发规范

我们在做的是一个前后端分离项目、需要通过接口文档对接的项目. 所以开发过程中要仔细查看前端所需的api接口和参数字段

为了严格按照接口进行开发，提高效率，对请求及响应格式进行规范化。

开发规范

2、post请求时有三种数据格式，可以提交form表单数据和Json数据（ContentType=application/json），文件等多部件类型（multipart/form-data）三种数据格式。jsonl类型的数据Servlet中使用fastjson进行解析

3、响应结果统一格式为json

开发规范

1、get 请求时，采用key/value格式请求，Servlet中可以使用getParameter() 获取。

2、post请求时有三种数据格式

第一种: Json数据,jsonl类型的数据Servlet中使用fastjson进行解析

第二种: 提交form表单数据

第三种: 文件等多部件类型（multipart/form-data）

3、响应结果统一格式为json

为什么使用JSON?

数据格式比较简单, 易于读写, JSON格式能够直接为服务器端代码使用, 大大简化了服务器端和客户端的代码开发量, 但是完成的任务不变, 且易于维护

本项目使用的是JSON解析工具为阿里巴巴的fastjson, maven工程导入下面的依赖即可.

```
<dependency>
  <groupId>com.alibaba</groupId>
  <artifactId>fastjson</artifactId>
  <version>1.1.37</version>
</dependency>

<dependency>
  <groupId>com.colobu</groupId>
  <artifactId>fastjson-jaxrs-json-provider</artifactId>
  <version>0.3.1</version>
</dependency>
```

2.2.3.2 接口文档

前端的开发基于服务端编写的接口，如果前端人员等待服务端人员将接口开发完毕再去开发前端内容这样做效率是非常低下的，所以当接口定义完成，可以使用工具生成接口文档，前端人员查看接口文档即可进行前端开发，这样前端和服务人员并行开发，大大提高了生产效率。



课程管理模块接口文档.md

2.2.3.3 编写CourseServlet

在CourseServlet中添加 **findCourseList**方法

```
@WebServlet("/course")
public class CourseServlet extends BaseServlet {

    //查询课程信息列表
    public void findCourseList(HttpServletRequest request, HttpServletResponse
response){

        try {
            //1.接收参数

            //2.业务处理
            CourseService cs = new CourseServiceImpl();
            List<Course> courseList = cs.findCourseList();

            //3.响应结果
            //SimplePropertyPreFilter 指定要转换的JSON字段
            SimplePropertyPreFilter filter = new
SimplePropertyPreFilter(Course.class,
                "id","course_name","price","sort_num","status");

            String result = JSON.toJSONString(courseList,filter);
            response.getWriter().print(result);

        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

2.3 Postman

2.3.1 postMan介绍

Postman是一款功能强大的http接口测试工具，使用postman可以完成http各种请求的功能测试。

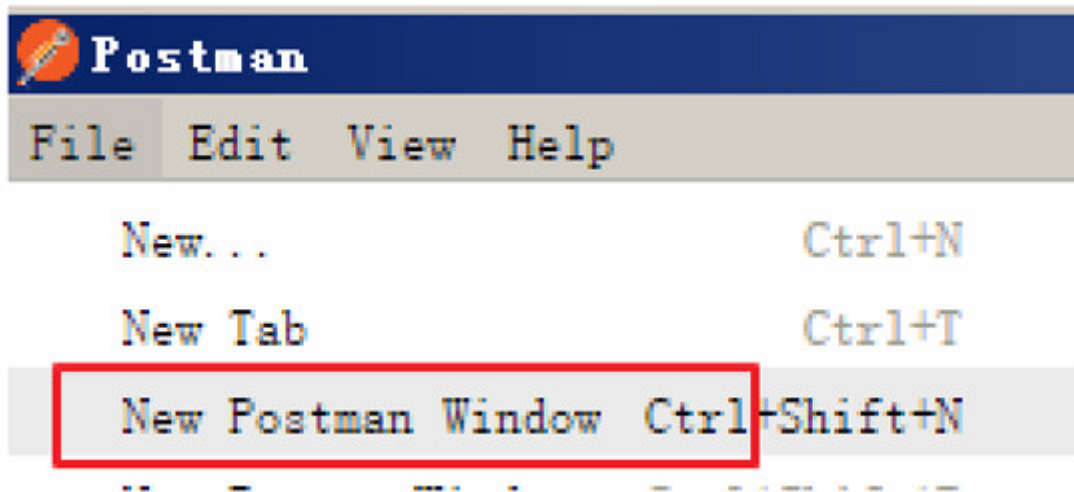
官方地址: <https://www.getpostman.com/>

安装Postman

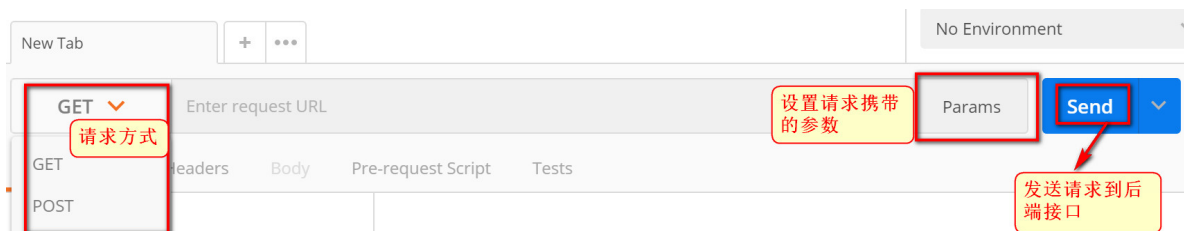
本教程使用，双击打开 Postman-win64-6.0.10-Setup.exe

2.3.2 Postman使用

1. 新建一个Postman窗口



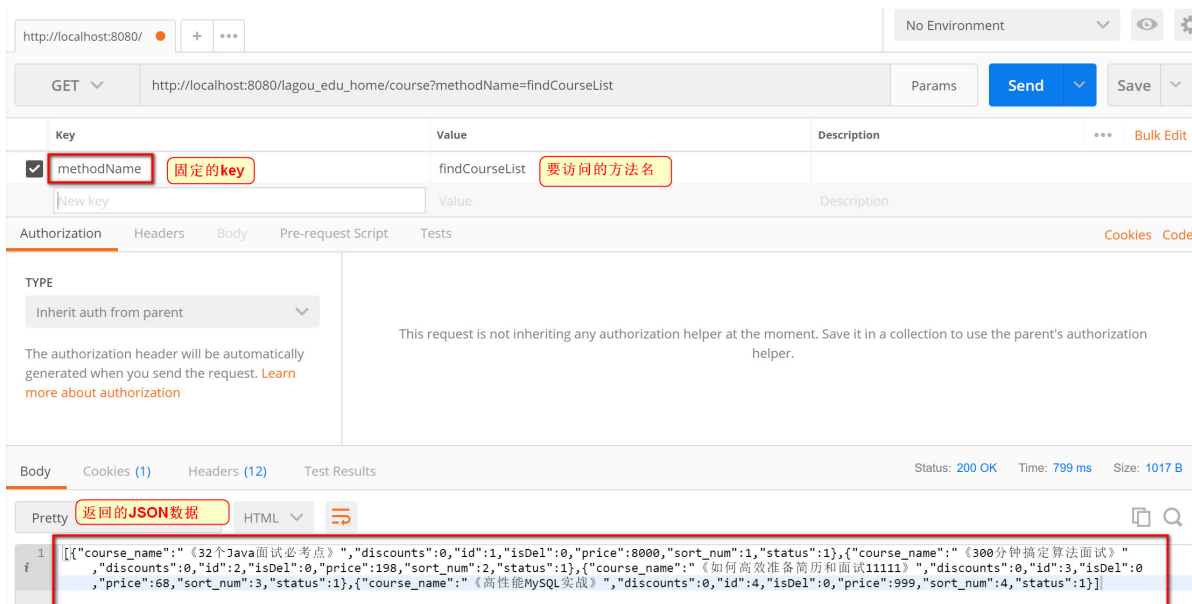
2. 窗口介绍



2.3.3 使用postman测试接口

1. 发送请求到指定的

`http://localhost:8080/lagou_edu_home/course?methodName=findCourseList`



2.3.4 创建模块将请求分类

1. 创建课程模块

Collections

+

CREATE A NEW COLLECTION

创建一个模块

Name

Collection Name

Name

cousese

名称

Description

Authorization

Pre-request Scripts

Tests

Variables

This description will show in your collection's documentation, along with the descriptions of its folders and requests.

课程管理模块

描述

Descriptions support Markdown

Cancel

Create

2. 选择 Save As 将请求保存到对应模块中

课程信息页面展示

Examples (0)

GET

http://localhost:8080/lagou_edu_home/course?methodName=findCourseList

Params

Send

Save

Authorization

Headers

Body

Pre-request Script

Tests

将上面的这个URL请求保存到对应的文件夹

Save As...

3. 描述一下请求的相关信息

Request name

http://localhost:8080/lagou_edu_home/course?methodName=findCour

Request description (Optional)

获取课程列表信息

请求描述信息

Descriptions support Markdown

Select a collection or folder to save to:

Search for a collection or folder

< course

+ Create Folder

GET http://localhost:8080/lagou_edu_home/course?methodName=fi...

保存到course中

Cancel

Save to course

3. 功能二: 多条件查询课程信息

3.1 需求分析

1. 根据课程名称和课程状态进行查询

课程名称 全部 状态 全部 查询

2. 要查询的字段

id, course_name, price, sort_num, STATUS

3. 查询条件

```
is_del  
course_name  
statuts
```

3.2 根据条件查询课程信息

3.2.2 Dao层编写

1. 因为是多条件查询,所以要注意多个参数情况下,SQL的编写

接口

```
/**  
 * 根据课程名称,课程状态 查询课程信息  
 * */  
public List<Course> findByCourseNameAndStatus(String courseName, String  
status);
```

实现类

```
/**  
 * 根据课程名称,课程状态 查询课程信息  
 * */  
//根据条件查询课程信息  
@Override  
public List<Course> findByCourseNameAndStatus(String courseName, String  
status) {  
  
    try {  
        //1.创建QueryRunner  
        QueryRunner qr = new QueryRunner(DruidUtils.getDataSource());  
  
        //2.编写SQL 当前的查询为多条件不定项查询  
        //2.1 创建StringBuffer 对象,将SQL字符串 添加进缓冲区  
        StringBuffer sb = new StringBuffer("SELECT  
id,course_name,price,sort_num,STATUS FROM course WHERE 1=1 and is_del = ? ");  
  
        //2.2 创建list集合 保存参数  
        List<Object> list = new ArrayList<>();  
        list.add(0);  
  
        //2.3 判断传入的参数是否为空  
        if(courseName != null && courseName != ""){  
            sb.append(" AND course_name LIKE ?");  
            //like查询 需要拼接 %  
            courseName = "%" + courseName + "%";  
            //将条件放进list集合  
            list.add(courseName);  
        }  
  
        if(status != null && status != ""){  
            sb.append("AND STATUS = ?");  
            //将status 转换为 int  
            int i = Integer.parseInt(status);  
            list.add(i);  
        }  
    }  
}
```

```

        //执行查询
        List<Course> courseList = qr.query(sb.toString(), new
        BeanListHandler<Course>(Course.class), list.toArray());

        //返回结果
        return courseList;
    } catch (SQLException e) {
        e.printStackTrace();
        return null;
    }
}

```

3.2.3 Service层编写

CourseService 接口

```

public List<Course> findByCourseNameAndStatus(String courseName, String
status);

```

CourseServiceImpl 实现类

```

@Override
public List<Course> findByCourseNameAndStatus(String courseName, String
status) {

    return courseDao.findByCourseNameOrStatus(courseName,status);
}

```

3.2.4 Servlet编写

在CourseServlet中添加 **findByCourseNameOrStatus**方法

```

//根据条件查询课程信息
public void findByCourseNameOrStatus(HttpServletRequest request ,
HttpServletRequest response){

    try {
        //1.接收参数
        String courseName = request.getParameter("course_name");
        String status = request.getParameter("status");

        //2.业务处理
        CourseService cs = new CourseServiceImpl();
        List<Course> courseList = cs.findByCourseNameOrStatus(courseName,
status);

        //3.返回结果 响应JSON格式数据
        //使用 SimplePropertyPreFilter,指定要转换为JSON的字段
        SimplePropertyPreFilter filter =
            new
SimplePropertyPreFilter(Course.class,"id","course_name","price","sort_num","stat
us");

```

```

        String result = JSON.toJSONString(courseList, filter);
        response.getWriter().println(result);

    } catch (IOException e) {
        e.printStackTrace();
    }
}

```

3.2.5 接口测试

- 请查阅接口文档,使用postman进行接口测试.

4. 功能三: 新建课程营销信息

4.1 需求分析

1. 选择新建课程,对课程营销信息进行录入

The screenshot shows a web interface for course management. At the top left, there is a blue button labeled '+ 新建课程' (New Course). To its right is a search bar with '课程名称' (Course Name) and a dropdown menu set to '全部' (All). Further right is a '状态' (Status) dropdown also set to '全部', followed by a '查询' (Search) button. A red box highlights the '+ 新建课程' button, and a yellow callout box points to it with the text: '新建课程就是在配置课程的营销信息,是一个插入操作' (New course is configuring the marketing information of the course, it is an insert operation). Below the search bar is a table with the following columns: ID, 课程名称 (Course Name), 价格 (Price), 排序 (Sort), 状态 (Status), and 操作 (Operations). The table contains one row with ID 3, Course Name 'Flutter实战与进阶', Price '1/98', Sort '0', and Status '下架' (Offline). The '操作' column for this row contains three links: '上架' (Shelve), '营销信息' (Marketing Information), and '配置课时' (Configure Class Hours).

4.1.1 基本信息

The form for '基本信息' (Basic Information) contains the following fields:

- * 名称:** 最多允许输入50个字 **course_name**
- * 简介:** 最多允许输入100个字 **brief**
- * 讲师名称:** 最多允许输入20个字 **teacher_name**
- * 讲师介绍:** 最多允许输入50个字 **teacher_info**
- * 课程概述:** This field is split into two parts:
 - preview_first_field**: 最多允许输入20个字
 - preview_second_field**: 最多允最多允许输入20个字

4.1.2 销售信息

销售信息

* 售卖价格:

输入售卖价格 discounts

元

商品原价:

页面划线价格 price

元

活动文案:

促销文案, 最多4个字 price_tag

4.1.3 分享信息

分享信息

* 分享小图:

选择

链接: share_image_title

尺寸200*200px, png、jpg格式

* 标题:

最多允许输入40个字share_title

* 简介:

最多允许输入60个字 share_description

4.1.4 课程详情

课程详情 course_description

售卖页

详情页

清除全部内容

H B I 5 - 44 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 408 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 521 522 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539 540 541 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558 559 560 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576 577 578 579 580 581 582 583 584 585 586 587 588 589 590 591 592 593 594 595 596 597 598 599 600 601 602 603 604 605 606 607 608 609 610 611 612 613 614 615 616 617 618 619 620 621 622 623 624 625 626 627 628 629 630 631 632 633 634 635 636 637 638 639 640 641 642 643 644 645 646 647 648 649 650 651 652 653 654 655 656 657 658 659 660 661 662 663 664 665 666 667 668 669 670 671 672 673 674 675 676 677 678 679 680 681 682 683 684 685 686 687 688 689 690 691 692 693 694 695 696 697 698 699 700 701 702 703 704 705 706 707 708 709 710 711 712 713 714 715 716 717 718 719 720 721 722 723 724 725 726 727 728 729 730 731 732 733 734 735 736 737 738 739 740 741 742 743 744 745 746 747 748 749 750 751 752 753 754 755 756 757 758 759 760 761 762 763 764 765 766 767 768 769 770 771 772 773 774 775 776 777 778 779 780 781 782 783 784 785 786 787 788 789 790 791 792 793 794 795 796 797 798 799 800 801 802 803 804 805 806 807 808 809 810 811 812 813 814 815 816 817 818 819 820 821 822 823 824 825 826 827 828 829 830 831 832 833 834 835 836 837 838 839 840 841 842 843 844 845 846 847 848 849 850 851 852 853 854 855 856 857 858 859 860 861 862 863 864 865 866 867 868 869 870 871 872 873 874 875 876 877 878 879 880 881 882 883 884 885 886 887 888 889 890 891 892 893 894 895 896 897 898 899 900 901 902 903 904 905 906 907 908 909 910 911 912 913 914 915 916 917 918 919 920 921 922 923 924 925 926 927 928 929 930 931 932 933 934 935 936 937 938 939 940 941 942 943 944 945 946 947 948 949 950 951 952 953 954 955 956 957 958 959 960 961 962 963 964 965 966 967 968 969 970 971 972 973 974 975 976 977 978 979 980 981 982 983 984 985 986 987 988 989 990 991 992 993 994 995 996 997 998 999 1000 1001 1002 1003 1004 1005 1006 1007 1008 1009 1010 1011 1012 1013 1014 1015 1016 1017 1018 1019 1020 1021 1022 1023 1024 1025 1026 1027 1028 1029 1030 1031 1032 1033 1034 1035 1036 1037 1038 1039 1040 1041 1042 1043 1044 1045 1046 1047 1048 1049 1050 1051 1052 1053 1054 1055 1056 1057 1058 1059 1060 1061 1062 1063 1064 1065 1066 1067 1068 1069 1070 1071 1072 1073 1074 1075 1076 1077 1078 1079 1080 1081 1082 1083 1084 1085 1086 1087 1088 1089 1090 1091 1092 1093 1094 1095 1096 1097 1098 1099 1100 1101 1102 1103 1104 1105 1106 1107 1108 1109 1110 1111 1112 1113 1114 1115 1116 1117 1118 1119 1120 1121 1122 1123 1124 1125 1126 1127 1128 1129 1130 1131 1132 1133 1134 1135 1136 1137 1138 1139 1140 1141 1142 1143 1144 1145 1146 1147 1148 1149 1150 1151 1152 1153 1154 1155 1156 1157 1158 1159 1160 1161 1162 1163 1164 1165 1166 1167 1168 1169 1170 1171 1172 1173 1174 1175 1176 1177 1178 1179 1180 1181 1182 1183 1184 1185 1186 1187 1188 1189 1190 1191 1192 1193 1194 1195 1196 1197 1198 1199 1200 1201 1202 1203 1204 1205 1206 1207 1208 1209 1210 1211 1212 1213 1214 1215 1216 1217 1218 1219 1220 1221 1222 1223 1224 1225 1226 1227 1228 1229 1230 1231 1232 1233 1234 1235 1236 1237 1238 1239 1240 1241 1242 1243 1244 1245 1246 1247 1248 1249 1250 1251 1252 1253 1254 1255 1256 1257 1258 1259 1260 1261 1262 1263 1264 1265 1266 1267 1268 1269 1270 1271 1272 1273 1274 1275 1276 1277 1278 1279 1280 1281 1282 1283 1284 1285 1286 1287 1288 1289 1290 1291 1292 1293 1294 1295 1296 1297 1298 1299 1300 1301 1302 1303 1304 1305 1306 1307 1308 1309 1310 1311 1312 1313 1314 1315 1316 1317 1318 1319 1320 1321 1322 1323 1324 1325 1326 1327 1328 1329 1330 1331 1332 1333 1334 1335 1336 1337 1338 1339 1340 1341 1342 1343 1344 1345 1346 1347 1348 1349 1350 1351 1352 1353 1354 1355 1356 1357 1358 1359 1360 1361 1362 1363 1364 1365 1366 1367 1368 1369 1370 1371 1372 1373 1374 1375 1376 1377 1378 1379 1380 1381 1382 1383 1384 1385 1386 1387 1388 1389 1390 1391 1392 1393 1394 1395 1396 1397 1398 1399 1400 1401 1402 1403 1404 1405 1406 1407 1408 1409 1410 1411 1412 1413 1414 1415 1416 1417 1418 1419 1420 1421 1422 1423 1424 1425 1426 1427 1428 1429 1430 1431 1432 1433 1434 1435 1436 1437 1438 1439 1440 1441 1442 1443 1444 1445 1446 1447 1448 1449 1450 1451 1452 1453 1454 1455 1456 1457 1458 1459 1460 1461 1462 1463 1464 1465 1466 1467 1468 1469 1470 1471 1472 1473 1474 1475 1476 1477 1478 1479 1480 1481 1482 1483 1484 1485 1486 1487 1488 1489 1490 1491 1492 1493 1494 1495 1496 1497 1498 1499 1500 1501 1502 1503 1504 1505 1506 1507 1508 1509 1510 1511 1512 1513 1514 1515 1516 1517 1518 1519 1520 1521 1522 1523 1524 1525 1526 1527 1528 1529 1530 1531 1532 1533 1534 1535 1536 1537 1538 1539 1540 1541 1542 1543 1544 1545 1546 1547 1548 1549 1550 1551 1552 1553 1554 1555 1556 1557 1558 1559 1560 1561 1562 1563 1564 1565 1566 1567 1568 1569 1570 1571 1572 1573 1574 1575 1576 1577 1578 1579 1580 1581 1582 1583 1584 1585 1586 1587 1588 1589 1590 1591 1592 1593 1594 1595 1596 1597 1598 1599 1600 1601 1602 1603 1604 1605 1606 1607 1608 1609 1610 1611 1612 1613 1614 1615 1616 1617 1618 1619 1620 1621 1622 1623 1624 1625 1626 1627 1628 1629 1630 1631 1632 1633 1634 1635 1636 1637 1638 1639 1640 1641 1642 1643 1644 1645 1646 1647 1648 1649 1650 1651 1652 1653 1654 1655 1656 1657 1658 1659 1660 1661 1662 1663 1664 1665 1666 1667 1668 1669 1670 1671 1672 1673 1674 1675 1676 1677 1678 1679 1680 1681 1682 1683 1684 1685 1686 1687 1688 1689 1690 1691 1692 1693 1694 1695 1696 1697 1698 1699 1700 1701 1702 1703 1704 1705 1706 1707 1708 1709 1710 1711 1712 1713 1714 1715 1716 1717 1718 1719 1720 1721 1722 1723 1724 1725 1726 1727 1728 1729 1730 1731 1732 1733 1734 1735 1736 1737 1738 1739 1740 1741 1742 1743 1744 1745 1746 1747 1748 1749 1750 1751 1752 1753 1754 1755 1756 1757 1758 1759 1760 1761 1762 1763 1764 1765 1766 1767 1768 1769 1770 1771 1772 1773 1774 1775 1776 1777 1778 1779 1780 1781 1782 1783 1784 1785 1786 1787 1788 1789 1790 1791 1792 1793 1794 1795 1796 1797 1798 1799 1800 1801 1802 1803 1804 1805 1806 1807 1808 1809 1810 1811 1812 1813 1814 1815 1816 1817 1818 1819 1820 1821 1822 1823 1824 1825 1826 1827 1828 1829 1830 1831 1832 1833 1834 1835 1836 1837 1838 1839 1840 1841 1842 1843 1844 1845 1846 1847 1848 1849 1850 1851 1852 1853 1854 1855 1856 1857 1858 1859 1860 1861 1862 1863 1864 1865 1866 1867 1868 1869 1870 1871 1872 1873 1874 1875 1876 1877 1878 1879 1880 1881 1882 1883 1884 1885 1886 1887 1888 1889 1890 1891 1892 1893 1894 1895 1896 1897 1898 1899 1900 1901 1902 1903 1904 1905 1906 1907 1908 1909 1910 1911 1912 1913 1914 1915 1916 1917 1918 1919 1920 1921 1922 1923 1924 1925 1926 1927 1928 1929 1930 1931 1932 1933 1934 1935 1936 1937 1938 1939 1940 1941 1942 1943 1944 1945 1946 1947 1948 1949 1950 1951 1952 1953 1954 1955 1956 1957 1958 1959 1960 1961 1962 1963 1964 1965 1966 1967 1968 1969 1970 1971 1972 1973 1974 1975 1976 1977 1978 1979 1980 1981 1982 1983 1984 1985 1986 1987 1988 1989 1990 1991 1992 1993 1994 1995 1996 1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020 2021 2022 2023 2024 2025 2026 2027 2028 2029 2030 2031 2032 2033 2034 2035 2036 2037 2038 2039 2040 2041 2042 2043 2044 2045 2046 2047 2048 2049 2050 2051 2052 2053 2054 2055 2056 2057 2058 2059 2060 2061 2062 2063 2064 2065 2066 2067 2068 2069 2070 2071 2072 2073 2074 2075 2076 2077 2078 2079 2080 2081 2082 2083 2084 2085 2086 2087 2088 2089 2090 2091 2092 2093 2094 2095 2096 2097 2098 2099 2100 2101 2102 2103 2104 2105 2106 2107 2108 2109 2110 2111 2112 2113 2114 2115 2116 2117 2118 2119 2120 2121 2122 2123 2124 2125 2126 2127 2128 2129 2130 2131 2132 2133 2134 2135 2136 2137 2138 2139 2140 2141 2142 2143 2144 2145 2146 2147 2148 2149 2150 2151 2152 2153 2154 2155 2156 2157 2158 2159 2160 2161 2162 2163 2164 2165 2166 2167 2168 2169 2170 2171 2172 2173 2174 2175 2176 2177 2178 2179 2180 2181 2182 2183 2184 2185 2186 2187 2188 2189 2190 2191 2192 2193 2194 2195 2196 2197 2198 2199 2200 2201 2202 2203 2204 2205 2206 2207 2208 2209 2210 2211 2212 2213 2214 2215 2216 2217 2218 2219 2220 2221 2222 2223 2224 2225 2226 2227 2228 2229 2230 2231 2232 2233 2234 2235 2236 2237 2238 2239 2240 2241 2242 2243 2244 2245 2246 2247 2248 2249 2250 2251 2252 2253 2254 2255 2256 2257 2258 2259 2260 2261 2262 2263 2264 2265 2266 2267 2268 2269 2270 2271 2272 2273 2274 2275 2276 2277 2278 2279 2280 2281 2282 2283 2284 2285 2286 2287 2288 2289 2290 2291 2292 2293 2294 2295 2296 2297 2298 2299 2300 2301 2302 2303 2304 2305 2306 2307 2308 2309 2310 2311 2312 2313 2314 2315 2316 2317 2318 2319 2320 2321 2322 2323 2324 2325 2326 2327 2328 2329 2330 2331 2332 2333 2334 2335 2336 2337 2338 2339 2340 2341 2342 2343 2344 2345 2346 2347 2348 2349 2350 2351 2352 2353 2354 2355 2356 2357 2358 2359 2360 2361 2362 2363 2364 2365 2366 2367 2368 2369 2370 2371 2372 2373 2374 2375 2376 2377 2378 2379 2380 2381 2382 2383 2384 2385 2386 2387 2388 2389 2390 2391 2392 2393 2394 2395 2396 2397 2398 2399 2400 2401 2402 2403 2404 2405 2406 2407 2408 2409 2410 2411 2412 2413 2414 2415 2416 2417 2418 2419 2420 2421 2422 2423 2424 2425 2426 2427 2428 2429 2430 2431 2432 2433 2434 2435 2436 2437 2438 2439 2440 2441 2442 2443 2444 2445 2446 2447 2448 2449 2450 2451 2452 2453 2454 2455 2456 2457 2458 2459 2460 2461 2462 2463 2464 2465 2466 2467 2468 2469 2470 2471 2472 2473 2474 2475 2476 2477 2478 2479 2480 2481 2482 2483 2484 2485 2486 2487 2488 2489 2490 2491 2492 2493 2494 2495 2496 2497 2498 2499 2500 2501 2502 2503 2504 2505 2506 2507 2508 2509 2510 2511 2512 2513 2514 2515 2516 2517 2518 2519 2520 2521 2522 2523 2524 2525 2526 2527 2528 2529 2530 2531 2532 2533 2534 2535 2536 2537 2538 2539 2540 2541 2542 2543 2544 2545 2546 2547 2548 2549 2550 2551 2552 2553 2554 2555 2556 2557 2558 2559 2560 2561 2562 2563 2564 2565 2566 2567 2568 2569 2570 2571 2572 2573 2574 2575 2576 2577 2578 2579 2580 2581 2582 2583 2584 2585 2586 2587 2588 2589 2590 2591 2592 2593 2594 2595 2596 2597 2598 2599 2600 2601 2602 2603 2604 2605 2606 2607 2608 2609 2610 2611 2612 2613 2614 2615 2616 2617 2618 2619 2620 2621 2


```

        "course_name,\n" +
        "brief,\n" +
        "teacher_name,\n" +
        "teacher_info,\n" +
        "preview_first_field,\n" +
        "preview_second_field,\n" +
        "discounts,\n" +
        "price,\n" +
        "price_tag,\n" +
        "share_image_title,\n" +
        "share_title,\n" +
        "share_description,\n" +
        "course_description,\n" +
        "course_img_url,\n" +
        "STATUS,\n" +
        "create_time,\n" +
        "update_time\n" +
        ")VALUES(?,?,?,?,?,?,?,?,?,?,?,?,?,?,?);";

//3.准备参数
Object[] param =
{course.getCourse_name(),course.getBrief(),course.getTeacher_name(),course.getTeacher_info(),

    course.getPreview_first_field(),course.getPreview_second_field(),course.getDiscounts(),course.getPrice(),

    course.getPrice_tag(),course.getShare_image_title(),course.getShare_title(),course.getShare_description(),

    course.getCourse_description(),course.getCourse_img_url(),course.getStatus(),course.getCreate_time(),course.getUpdate_time()};

//4.执行插入操作
int row = qr.update(sql, param);

    return row;
} catch (SQLException e) {
    e.printStackTrace();
    return 0;
}
}

```

4.3 Dao层方法测试

```

//测试保存课程营销信息
@Test
public void testSaveCourseSalesInfo(){

    //1.创建course对象
    Course course = new Course();
    course.setCourse_name("爱情36计");
    course.setBrief("学会去找对象");
    course.setTeacher_name("药水哥");
    course.setTeacher_info("人人都是药水哥");
    course.setPreview_first_field("共10讲");
}

```

```

        course.setPreview_second_field("每周日更新");
        course.setDiscounts(88.88);
        course.setPrice(188.0);
        course.setPrice_tag("最新优惠价");
        course.setShare_image_title("哈哈哈");
        course.setShare_title("嘻嘻嘻");
        course.setShare_description("天天向上");
        course.setCourse_description("爱情36计,就像一场游戏");
        course.setCourse_img_url("https://www.xx.com/xxx.jpg");
        course.setStatus(1); //1 上架 ,0 下架
        String formart = DateUtils.getDateFormart();
        course.setCreate_time(formart);
        course.setUpdate_time(formart);

        int i = courseDao.saveCourseSalesInfo(course);
        System.out.println(i);
    }

```

4.4 Service层编写

1.编写枚举类,设置响应状态码

```

public enum StatusCode {

    SUCCESS(0, "success"),
    FAIL(1, "fail");

    //定义属性
    private int code;
    private String message;

    //定义构造
    StatusCode(int code, String message) {
        this.code = code;
        this.message = message;
    }

    //get/set
    public int getCode() {
        return code;
    }

    public void setCode(int code) {
        this.code = code;
    }

    public String getMessage() {
        return message;
    }

    public void setMessage(String message) {
        this.message = message;
    }
}

```

```
//重写toString,将枚举对象转化为JSON
@Override
public String toString() {
    JSONObject object = new JSONObject();
    object.put("status",code);
    object.put("msg",message);
    return object.toString();
}
}
```

2. 编写Service

接口

```
public String saveCoursesSalesInfo(Course course);
```

实现类

```
@Override
public String saveCoursesSalesInfo(Course course) {
    //1. 补全课程信息
    String dateFormat = DateUtils.getDateFormat();
    course.setCreate_time(dateFormat);
    course.setUpdate_time(dateFormat);
    course.setStatus(0);

    //2. 调用Dao进行插入
    int i = courseDao.saveCoursesSalesInfo(course);
    if(i > 0){
        //保存成功
        String result = StatusCode.SUCCESS.toString();
        return result;

    }else{
        //保存失败
        String result = StatusCode.FAIL.toString();
        return result;
    }
}
```

4.5 文件上传

4.5.1 图片上传分析

在添加课程营销信息的表单中,有一个图片上传项

* 分享小图:

选择

链接:

可输入图片链接

尺寸200*200px, png、jpg格式

4.5.2 文件上传介绍

文件上传的实质：文件的拷贝

- 文件上传：从本地将文件拷贝到服务器磁盘上
 - 客户端：需要编写文件上传表单
 - 服务端：需要编写代码接受上传的文件

4.5.3 客户端编码

1. 文件上传三要素：

- 1. 表单提交方式：**post** (get方式提交有大小限制, post没有)
- 2. 表单的enctype属性：必须设置为 **multipart/form-data**.
 - enctype就是encoding就是编码类型的意思。
 - multipart/form-data是多部件文件上传，指表单数据有多部分构成，既有文本数据，又有文件等二进制数据的意思。
- 3. 表单必须有文件上传项：**file**，必须要有name属性和值

1. 提交方式必须是POST

```
<form action="" method="post" enctype="multipart/form-data">  
  <input type="file" name="logoImage"/><br/>  
  <input type="submit" value="文件上传"/>  
</form>
```

2. 表单的enctype类型 必须是 multipart/form-data

3. input的type类型必须是 file name属性也必须写

注意：默认情况下，表单的enctype的值是application/x-www-form-urlencoded，不能用于文件上传，只有使用了multipart/form-data，才能完整的传递文件数据

2. 代码示例

```
<%@ page contentType="text/html; charset=UTF-8" language="java" %>  
<html>  
<head>  
  <title>Title</title>  
</head>  
<body>  
  <!--  
    表单提交必须是POST，  
    表单的enctype属性：必须设置为 multipart/form-data。  
    input的type类型必须指定为：file，一定要有name属性  
  --%>  
  <form action="${pageContext.request.contextPath}/upload" method="post"  
    enctype="multipart/form-data">  
    <input type="file" name="upload">  
    <br>  
    <input type="text" name="name">  
    <input type="text" name="password">  
    <input type="submit" value="文件上传">  
  </form>
```

```
</body>
</html>
```

4.5.4 服务端编码

服务端要接收文件上传的表单数据

1. 上传文件, 抓包分析

使用360浏览器进行抓包,谷歌浏览器不方便查看

文件上传的接收原理

▼ Request Headers view source

Content-Type: multipart/form-data; boundary=----WebKitFormBoundary3DfBw6vqOdoveTPQ

请求头

表单项之间的分隔符

请求体

▼ Request Payload 表单内容被分隔成了三部分

```
-----WebKitFormBoundary3DfBw6vqOdoveTPQ
Content-Disposition: form-data; name="upload"; filename="aaa.txt"
Content-Type: text/plain
```

```
-----WebKitFormBoundary3DfBw6vqOdoveTPQ
Content-Disposition: form-data; name="name"
```

```
536
-----WebKitFormBoundary3DfBw6vqOdoveTPQ
Content-Disposition: form-data; name="password"
```

```
6436
-----WebKitFormBoundary3DfBw6vqOdoveTPQ--
```

2. 服务端获上传的文件

1. 通过request获取请求体的内容
2. 解析请求体 多部件上传的特点是,每个input都是一个表单项.

根据分隔符将请求中所有的内容,切割成数组,数组中的每一个元素 都是一个表单项

3. 遍历数组,分清楚那个是普通的表单项, 哪个是 文件上传项

如何区分? 判断是否有 filename

4. 获取到普通表单项中的内容,通过属性name获取

5. 获取文件上传项内容

文件名: filename = aaa.txt

文件内容:

6. 使用IO将文件内容,保存到服务器中

4.5.5 FileUpload工具类

1. 导入依赖

FileUpload包可以很容易地将文件上传到你的Web应用程序.

IOUtils封装了Java中io的常见操作，使用十分方便，需要下载 commons-io-1.4.jar 包

```
<dependency>
  <groupId>commons-io</groupId>
  <artifactId>commons-io</artifactId>
  <version>1.4</version>
</dependency>

<dependency>
  <groupId>commons-fileupload</groupId>
  <artifactId>commons-fileupload</artifactId>
  <version>1.2.1</version>
</dependency>
```

2. FileUpload 核心类介绍

类名	介绍
DiskFileItemFactory	磁盘文件项工厂，读取文件时相关的配置,比如: 缓存的大小，临时目录的位置
ServletFileUpload	文件上传的一个核心类
FileItem	代表每一个表单项

3. 文件上传的API的详解

- ServletFileUpload

方法	说明
isMultipartContent(request);	判断是否是一个文件上传的表单
parseRequest(request);	解析request获得表单项的集合
setHeaderEncoding("UTF-8");	设置上传的文件名的编码方式

- FileItem

方法	说明
isFormField()	判断是否是普通表单项
getFieldName()	获得表单的name属性值
item.getString()	获得表单的value值
getName()	获得上传文件的名称
getInputStream()	获得上传文件
delete()	删除临时文件

4. 文件上传后台代码编写

FileUpload使用步骤:

- 1、创建磁盘文件项工厂
- 2、创建文件上传的核心类
- 3、解析request---获得文件项集合
- 4、遍历文件项集合
- 5、判断普通表单项/文件上传项

```
@WebServlet("/upload")
public class FileUploadServlet extends HttpServlet {

    @Override
    protected void doGet(HttpServletRequest req, HttpServletResponse resp)
        throws ServletException, IOException {

        try {
            //1.创建磁盘文件项工厂
            DiskFileItemFactory factory = new DiskFileItemFactory();

            //2.创建文件上传核心类
            ServletFileUpload upload = new ServletFileUpload(factory);
            //2.1 设置上传文件名的编码
            upload.setHeaderEncoding("utf-8");
            //2.2 判断表单是否是文件上传表单
            boolean multipartContent = upload.isMultipartContent(req);
            //2.3 是文件上传表单
            if(multipartContent){

                //3. 解析request ,获取文件项集合
                List<FileItem> list = upload.parseRequest(req);

                if(list != null){
                    //4.遍历获取表单项
                    for (FileItem item : list) {
                        //5. 判断是不是一个普通表单项
                        boolean formField = item.isFormField();
                        if(formField){
                            //普通表单项, 当 enctype="multipart/form-data"时,
                            //request的getParameter()方法 无法获取参数
                            String fieldName = item.getFieldName();
                            String value = item.getString("utf-8");//设置编码
                            System.out.println(fieldName + "=" + value);
                        }else{

                            //文件上传项
                            //文件名
                            String fileName = item.getName();

                            //避免图片名重复 拼接UUID
                            String newFileName = UUIDUtils.getUUID()+"_"+

fileName;

                            //获取输入流
```

```

        InputStream in = item.getInputStream();

        //创建输出流 输出到H盘
        FileOutputStream fos = new
FileOutputStream("H:/upload/" +newFileName);

        //使用工具类IOUtils,copy文件
        IOUtils.copy(in, fos);

        //关闭流
        fos.close();
        in.close();

    }

}

}

} catch (FileUploadException e) {
    e.printStackTrace();
}

}

@Override
protected void doPost(HttpServletRequest req, HttpServletResponse resp)
throws ServletException, IOException {
    doGet(req, resp);
}

}

```

4.5.6 将图片上传到tomcat服务器

1. 将项目部署到webapps

将部署方式改变为 war模式,把项目部署在tomcat的webapps下

- idea中部署项目两种方式
 - war模式：将项目以war包的形式上传真实到服务器的webapps目录中；
 - war exploded模式：仅仅是目录的映射，就相当于tomcat在项目源文件夹中启动一样；



2.在webapps中创建upload目录

upload目录专门用来保存上传过来的图片

➤ software ➤ apache-tomcat-8.5.55 ➤ webapps

名称

docs

examples

host-manager

lagou_edu_home

manager

ROOT

upload

项目被部署到了这里

upload文件夹专门用来保存图片

3.修改代码,将图片上传到服务器

- 修改图片的输出路径
 1. 获取到项目的运行目录信息
 2. 截取到webapps的 目录路径
 3. 拼接输出路径,将图片保存到upload

```
@Override
protected void doGet(HttpServletRequest req, HttpServletResponse resp)
throws ServletException, IOException {

    try {
        //1.创建磁盘文件项工厂
        DiskFileItemFactory factory = new DiskFileItemFactory();

        //2.创建文件上传核心类
        ServletFileUpload upload = new ServletFileUpload(factory);
        //2.1 设置上传文件名的编码
        upload.setHeaderEncoding("utf-8");
        //2.2 判断表单是否是文件上传表单
        boolean multipartContent = upload.isMultipartContent(req);
        //2.3 是文件上传表单
        if(multipartContent){

            //3. 解析request ,获取文件项集合
            List<FileItem> list = upload.parseRequest(req);
```

```

        if(list != null){
            //4.遍历获取表单项
            for (FileItem item : list) {
                //5. 判断是不是一个普通表单项
                boolean formField = item.isFormField();
                if(formField){
                    //普通表单项，当 enctype="multipart/form-data"时，
                    request.getParameter()方法 无法获取参数
                    String fieldName = item.getFieldName();
                    String value = item.getString("utf-8");//设置编码
                    System.out.println(fieldName + "=" + value);
                }else{

                    //文件上传项
                    //文件名
                    String fileName = item.getName();
                    //避免图片名重复 拼接UUID
                    String newFileName = UUIDUtils.getUUID()+"_"+
fileName;

                    //获取上传文件的内容
                    InputStream in = item.getInputStream();

                    String path =
this.getServletContext().getRealPath("/");

                    //获取到 webapps路径
                    String webappsPath = path.substring(0,
path.indexOf("lagou_edu_home"));
                    OutputStream out = new
FileOutputStream(webappsPath+"/upload/"+newFileName);
                    //拷贝文件到服务器
                    IOUtils.copy(in,out);

                    out.close();
                    in.close();
                }
            }
        }
    } catch (FileUploadException e) {
        e.printStackTrace();
    }
}

```

4. 页面加载图片

将tomcat作为图片服务器使用时，存储上传的图片后,如果想要图片可以访问,需要在idea中进行配置:

1. 选择external source ---> 找到webapps目录下的的upload文件夹



2. 上传一张图片到服务器
3. 在项目内部页面加载图片

```

```

4. 也可以通过HTTP方式访问

```
http://localhost:8080/upload/abbd99891af442a8a9cb65848744452e_qiyu.jpg
```

4.6 BeanUtils工具类

1. 介绍

BeanUtils 是 Apache commons组件的成员之一，主要用于简化JavaBean封装数据的操作。**可以将一个表单提交的所有数据封装到JavaBean中。**

2. 导入依赖

```
<dependency>
  <groupId>commons-beanutils</groupId>
  <artifactId>commons-beanutils</artifactId>
  <version>1.8.3</version>
</dependency>
```

3. BeanUtils 对象常用方法

方法	描述
populate(Object bean, Map properties)	将Map数据封装到指定JavaBean中，一般用于将表单的所有数据封装到JavaBean
setProperty(Object obj,String name,Object value)	设置属性值
getProperty(Object obj,String name)	获得属性值

4. BeanUtils 使用测试

```
public class TestBeanUtils {
```

```

@Test
public void test01() throws InvocationTargetException,
IllegalAccessException, NoSuchMethodException {

    //1.创建course对象
    Course course = new Course();

    //2.创建Map
    Map<String,Object> map = new HashMap<>();

    //3.向map集合中添加数据，key要与course的属性名保持一致,value的数据类型与course的
    属性的类型保持一致
    map.put("id",1);
    map.put("course_name","大数据");
    map.put("brief","课程包含所有大数据流行的技术");
    map.put("teacher_name","周星星");
    map.put("teacher_info","非著名演员");

    //将map中的数据封装到 course中
    BeanUtils.populate(course,map);

    System.out.println(course.getId()+" " + course.getCourse_name() +" "
+course.getBrief()
+" " +course.getTeacher_name()+" " +course.getTeacher_info());

    //设置属性 获取属性
    BeanUtils.setProperty(course,"price",100.0);

    String price = BeanUtils.getProperty(course, "price");

    System.out.println(price);
}
}

```

4.7 Servlet编写

4.7.1 CourseSalesInfoServlet

创建CourseSalesInfoServlet类,继承HttpServlet,完成保存课程营销信息操作.

因为上传的信息包含文件信息,无法直接通过request直接获取参数,所以不能继承BaseServlet

```

@WebServlet("/courseSalesInfo")
public class CourseSalesInfoServlet extends HttpServlet {

    /**
     * 保存课程营销信息
     *      收集表单数据,封装到course对象中,将图片上传到tomcat服务器中
     */
    @Override
    protected void doGet(HttpServletRequest req, HttpServletResponse resp)
    throws ServletException, IOException {

        try {
            //1.创建Course对象

```

```

Course course = new Course();

//2.创建Map集合,用来收集数据
Map<String,Object> map = new HashMap<>();

//3.创建磁盘工厂对象
DiskFileItemFactory factory = new DiskFileItemFactory();

//4.文件上传核心对象
ServletFileUpload fileUpload = new ServletFileUpload(factory);

//5.解析request对象,获取表单项集合
List<FileItem> list = fileUpload.parseRequest(req);

//6.遍历集合 判断哪些是普通的表单项,那些是文件表单项
for (FileItem item : list) {

    boolean formField = item.isFormField();
    if(formField){
        //是普通表单项,获取表单项中的数据,保存到map
        String fieldName = item.getFieldName();
        String value = item.getString("UTF-8");
        System.out.println(fieldName + " " + value);
        //使用map收集数据
        map.put(fieldName,value);
    }else{
        //文件上传项
        //获取文件名
        String fileName = item.getName();
        String newFileName = UUIDUtils.getUUID()+"_"+fileName;

        //获取输入流
        InputStream in = item.getInputStream();

        //获取webapps的目录路径
        String realPath = this.getServletContext().getRealPath("/");
        String wabappsPath = realPath.substring(0,
realPath.indexOf("lagou_edu_home"));

        //创建输出流
        OutputStream out = new
FileOutputStream(wabappsPath+"/upload/" + newFileName);

        IOUtils.copy(in,out);
        out.close();
        in.close();

        //将图片路径进行保存
        map.put("course_img_url", Constants.LOCAL_URL+"/upload/" +
newFileName);
    }
}

//使用BeanUtils 将map中的数据封装到course对象
BeanUtils.populate(course,map);

String dateFormart = DateUtils.getDateFormart();
CourseService cs = new CourseServiceImpl();

```

```

        if(map.get("id") != null){
            //修改操作
            //补全信息
            course.setUpdate_time(dateFormart);//修改时间
            String result = cs.updateCourseSalesInfo(course);
            //响应结果
            resp.getWriter().print(result);

        }else{
            //新建操作
            //补全信息
            course.setCreate_time(dateFormart);//创建时间
            course.setUpdate_time(dateFormart);//修改时间
            course.setStatus(1); //上架
            String result = cs.saveCourseSalesInfo(course);
            //响应结果
            resp.getWriter().print(result);
        }

    } catch (Exception e) {
        e.printStackTrace();
    }

}

@Override
protected void doPost(HttpServletRequest req, HttpServletResponse resp)
throws ServletException, IOException {
    doGet(req, resp);
}
}

```

4.7.2 接口测试

postman测试上传文件

1. 接口地址填写正确
2. 将请求方式设置为POST
3. 需要上传文件, 设置Headers: "key":"Content-Type", "value":"multipart/form-data"

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> Content-Type	multipart/form-data	

4. Body选择form-data
5. key 右侧下拉选择file; value 点击Select Files选择文件, 按照接口文档,补全测试参数

Params Authorization Headers (1) Body ● Pre-request Script Tests

● none ● form-data ● x-www-form-urlencoded ● raw ● binary

	KEY	VALUE
<input checked="" type="checkbox"/>	file	琦玉.jpg X
<input checked="" type="checkbox"/>	course_name	开胯秘籍
<input checked="" type="checkbox"/>	brief	跟着雷教练学习开胯
<input checked="" type="checkbox"/>	teacher_name	雷教练
<input checked="" type="checkbox"/>	teacher_info	技术精湛安全驾驶30年
<input checked="" type="checkbox"/>	preview_first_field	共5讲
<input checked="" type="checkbox"/>	preview_second_field	每周二更新
<input checked="" type="checkbox"/>	discounts	88.8
<input checked="" type="checkbox"/>	price	800.0
<input checked="" type="checkbox"/>	price_tag	先到先得
<input checked="" type="checkbox"/>	share_image_title	hello word
<input checked="" type="checkbox"/>	share_title	IT修炼之路永无止境
<input checked="" type="checkbox"/>	share_description	金牌讲师带你了解最新最牛的技术让你的实力再次进阶!
<input checked="" type="checkbox"/>	course_description	十年编程两茫茫，工期短，需求长。千行代码，Bug何处藏。...

4.7.3 保存图片URL优化

1.创建常量类

```
public final class Constants {  
  
    //本地访问地址  
    public static final String LOCAL_URL = "http://localhost:8080";  
}
```

2.拼接图片URL

```
//将图片路径进行保存  
map.put("course_img_url", Constants.LOCAL_URL+"/upload/" + newFileName);
```

5. 功能四: 修改课程营销信息

5.1 需求分析

营销信息其实就是课程相关的信息, 操作的依然是 **course** 表. 我们通过点击营销信息按钮,进入到对应的课程营销信息页面,对原有信息进行修改.



5.2 Dao层编写

1. 通过上面的分析,首先要编写 根据课程ID查询课程信息,进行回显

接口

```
//根据课程ID 查询课程信息  
public Course findCourseById(int id);
```

实现类

```
//根据课程ID 查询课程营销信息  
@Override  
public Course findCourseById(int id) {  
    try {  
        QueryRunner qr = new QueryRunner(DruidUtils.getDataSource());  
  
        String sql = "SELECT \n" +  
            "id,\n" +  
            "course_name,\n" +  
            "brief,\n" +  
            "teacher_name,\n" +  
            "teacher_info,\n" +  
            "preview_first_field,\n" +  
            "preview_second_field,\n" +  
            "discounts,\n" +  
            "price,\n" +  
            "price_tag,\n" +  
            "course_img_url,\n" +  
            "share_image_title,\n" +  
            "share_title,\n" +  
            "share_description,\n" +  
            "course_description,\n" +  
            "STATUS\n" +
```



```

        "FROM course WHERE id = ?;";

        Course course = qr.query(sql, new BeanHandler<Course>(Course.class),
id);

        return course;

    } catch (SQLException e) {
        e.printStackTrace();
        return null;
    }

}

```

```

-- 根据ID查询课程信息SQL
SELECT
id,
course_name,
brief,
teacher_name,
teacher_info,
preview_first_field,
preview_second_field,
discounts,
price,
price_tag,
course_img_url,
share_image_title,
share_title,
share_description,
course_description,
STATUS
FROM course WHERE id = ?;

```

2. 编写修改课程营销信息的方法,将修改写入数据库

接口

```

//修改课程营销信息
public int updateCourseSalesInfo(Course course);

```

实现类

```

//修改课程营销信息
@Override
public int updateCourseSalesInfo(Course course) {

    try {
        QueryRunner qr = new QueryRunner(DruidUtils.getDataSource());

        String sql = "UPDATE course SET \n" +
            "course_name = ?,\n" +
            "brief = ?,\n" +
            "teacher_name = ?,\n" +
            "teacher_info = ?,\n" +
            "preview_first_field = ?,\n" +
            "preview_second_field = ?,\n" +
            "discounts = ?,\n" +
            "price = ?,\n" +

```

```

        "price_tag = ?,\n" +
        "share_image_title = ?,\n" +
        "share_title = ?,\n" +
        "share_description = ?,\n" +
        "course_description = ?,\n" +
        "course_img_url = ?,\n" +
        "update_time = ?\n" +
        "WHERE id = ?";

        Object[] param =
{course.getCourse_name(),course.getBrief(),course.getTeacher_name(),course.getTeacher_info(),

        course.getPreview_first_field(),course.getPreview_second_field(),course.getDiscounts(),course.getPrice(),course.getPrice_tag(),

        course.getShare_image_title(),course.getShare_title(),course.getShare_description(),course.getCourse_description(),
        course.getCourse_img_url(),course.getUpdate_time(),course.getId()};

        int row = qr.update(sql, param);
        return row;

    } catch (SQLException e) {
        e.printStackTrace();
        return 0;
    }

}

```

修改课程

```

UPDATE course SET
course_name = ?,
brief = ?,
teacher_name = ?,
teacher_info = ?,
preview_first_field = ?,
preview_second_field = ?,
discounts = ?,
price = ?,
price_tag = ?,
share_image_title = ?,
share_title = ?,
share_description = ?,
course_description = ?,
course_img_url = ?,
update_time = ?
WHERE id = ?

```

3. 测试

5.3 Service层编写

接口

```
public Course findCourseById(int id);
```

实现类

```
@Override
public Course findCourseById(int id) {
    return courseDao.findCourseById(id);
}
```

接口

```
public String updateCourseSalesInfo(Course course);
```

实现类

```
@Override
public String updateCourseSalesInfo(Course course) {

    //调用dao
    int i = courseDao.updateCourseSalesInfo(course);

    //根据插入是否成功,封装对应信息

    if(i > 0){
        //保存成功
        String result = StatusCode.SUCCESS.toString();
        return result;
    }else{
        //保存失败
        String result = StatusCode.FAIL.toString();
        return result;
    }

}
```

5.4 Servlet编写

5.4.1 根据ID查询课程信息

5.4.1.1 CourseServlet

在CourseServlet中, 添加根据ID查询课程信息的功能

```
/**
 * 根据课程ID查询课程营销信息
 * */
public void findCourseById(HttpServletRequest request , HttpServletResponse response){

    try {
        //1.接收参数
        String id = request.getParameter("id");

        //2.业务处理
        CourseService cs = new CourseServiceImpl();
        Course course = cs.findCourseById(Integer.parseInt(id));
    }
}
```

```

//3.返回结果 响应JSON格式数据
//使用 SimplePropertyPreFilter,指定要转换为JSON的字段
SimplePropertyPreFilter filter = new
SimplePropertyPreFilter(Course.class,"id","course_name","brief","teacher_name",

    "teacher_info","preview_first_field","preview_second_field","discounts","price"
    ,"price_tag","share_image_title","share_title","share_description","course_descr
    iption");

    String result = JSON.toJSONString(course, filter);
    response.getWriter().println(result);

} catch (IOException e) {
    e.printStackTrace();
}
}

```

5.4.1.2 接口测试

详见接口文档

5.4.2 修改CourseSalesInfoServlet

5.4.2.1 需求分析

保存营销信息和修改营销信息,访问的是同一个接口,所以在**CourseSalesInfoServlet**中,我们需要进行一下判断

- 携带id 就是修改操作
- 未携带id就是新增操作

5.4.2.2 代码修改

```

@WebServlet("/courseSalesInfo")
public class CourseSalesInfoServlet extends HttpServlet {

    /**
     * 保存营销信息
     * 收集表单的数据 封装一个Course实体 将上传图片存到服务器磁盘上
     */
    @Override
    protected void doGet(HttpServletRequest req, HttpServletResponse resp)
    throws ServletException, IOException {

        try {
            //1.获取参数,调用FileUploadUtils,进行文件上传和参数的封装
            Map<String, Object> map = FileUploadUtil.upload(req);

            //2.使用BeanUtils 将map中的数据封装到 Course对象中
            Course course = new Course();
            BeanUtils.populate(course,map);

            //3.业务处理
            if(map.get("id") != null){

```

```

        //补充信息 修改时间
        course.setUpdate_time(DateUtils.getDateFormart());

        CourseService cs = new CourseServiceImpl();
        Map<String, String> message = cs.updateSalesInfo(course);

        //4.响应JSON数据
        String result = JSON.toJSONString(message);
        resp.getWriter().println(result);

    }else{

        //补充信息
        course.setCreate_time(DateUtils.getDateFormart()); //创建时间
        course.setUpdate_time(DateUtils.getDateFormart()); //修改时间
        course.setStatus(0); //状态

        //8.业务处理
        CourseService cs = new CourseServiceImpl();
        Map<String, String> message = cs.saveSalesInfo(course);

        //9.响应JSON数据
        String result = JSON.toJSONString(message);
        resp.getWriter().println(result);
    }

    } catch (Exception e) {
        e.printStackTrace();
    }

}

@Override
protected void doPost(HttpServletRequest req, HttpServletResponse resp)
throws ServletException, IOException {
    doGet(req, resp);
}

}

```

5.4.2.3 接口测试

根据接口文档,进行测试

6. 功能五: 修改课程状态

6.1 需求分析

1. 数据库中课程状态码为0或者1,课程状态, 0-草稿(下架), 1-上架

status	int(2) NULL	课程状态, 0-草稿, 1-上架
--------	-------------	------------------

2. 页面分析

ID	课程名称	价格	排序	状态	点击上架,修改状态为上架		
3	Flutter实战与进阶	1/98	0	• 下架	上架	营销信息	配置课时

6.2 DAO层编写

接口

```
//修改课程状态
int updateCourseStatus(Course course);
```

实现类

```
//修改课程状态
@Override
public int updateCourseStatus(Course course) {

    try {
        QueryRunner qr = new QueryRunner(DruidUtils.getDataSource());

        String sql = "UPDATE course SET STATUS = ? ,update_time = ? WHERE id = ?";

        Object[] param =
{course.getStatus(),course.getUpdate_time(),course.getId()};

        int row = qr.update(sql, param);

        return row;
    } catch (SQLException e) {
        e.printStackTrace();
        return 0;
    }
}
```

6.3 Service层编写

接口

```
public Map<String,Integer> updateCourseStatus(Course course);
```

实现类

```
@Override
public Map<String, Integer> updateCourseStatus(Course course) {

    //调用dao
    int row = courseDao.updateCourseStatus(course);

    Map<String ,Integer> map = new HashMap<>();

    if(row > 0){

        if(course.getStatus() == 0){
            map.put("status",0);
        }
    }
}
```

```

        }else{
            map.put("status",1);
        }
    }

    return map;
}

```

6.4 Servlet编写

在CourseServlet中, 添加updateCourseStatus方法

```

//修改课程状态
public void updateCourseStatus(HttpServletRequest request,HttpServletResponse response){

    try {
        //1. 获取参数
        String id = request.getParameter("id");

        //2. 业务处理
        CourseService cs = new CourseServiceImpl();

        //3. 根据课程id 查询课程信息
        Course course = cs.findCourseById(Integer.parseInt(id));

        //4. 判断课程信息状态,进行取反设置
        int status = course.getStatus();
        if(status == 0){
            //如果是0 设置为1
            course.setStatus(1);
        }else{
            course.setStatus(0);
        }

        //5. 设置更新时间
        course.setUpdate_time(DateUtils.getDateFormart());

        //6. 修改状态
        Map<String, Integer> map = cs.updateCourseStatus(course);

        //7. 响应结果
        String result = JSON.toJSONString(map);

        response.getWriter().print(result);

    } catch (IOException e) {
        e.printStackTrace();
    }
}

```

6.5 接口测试

查看接口文档,进行测试