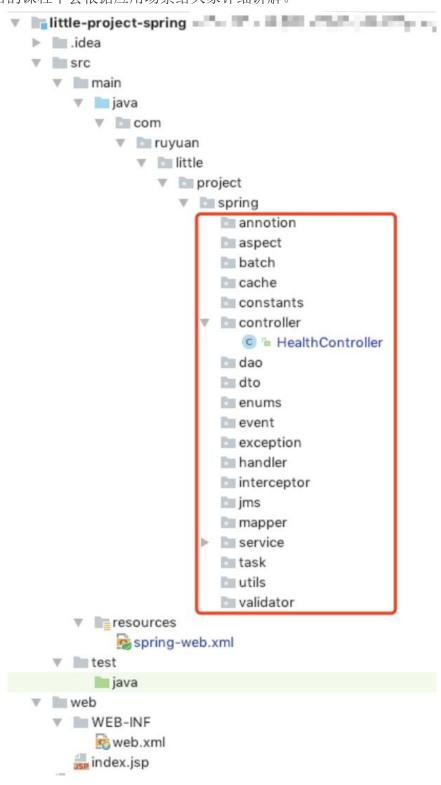
# 07\_有条不紊:对我们要做的项目进行整体划分

### 1、开篇

上节课我们对互联网教育系统进行了项目初始化,其中介绍了项目的创建方式、依赖项、配置、健康检查接口。无论什么项目都会包括不同的功能模块,这些模块相互独立,之间也存在关联。通常来说我们会讲这些模块单独创建成服务进行管理,为了方便教学的缘故在这个项目中将这些模块以包的形式在一个项目中进行管理。今天就将介绍这些模块的划分:

- 模块划分
- Annotation
- Aspect
- Batch
- Cache
- Constants
- Controller
- DAO
- DTO
- Enums
- Event
- Exception
- Handler
- Interceptor
- JMS
- Mapper
- Service
- Task
- Utils
- Validator
  - 2、模块划分

如图 1 所示,先通过一张大图了解一些整个项目的包结构,这些包都位于 com.ruyuan.little.project.spring 的 namesapce 之下,红框的部分就是本节课需要 介绍的包,这里我会带大家过一遍。大家可以在这个过程中建立对每个组件包的印象,随后的课程中会根据应用场景给大家详细讲解。



### 图 1 互联网教育系统模块划分

#### 3. Annotation

这个包下面主要包括定制化的标签,如图 2 所示,可以在自定义的 annotaion 中加入 IsDate 的 annotation。类型是针对 ElementType.FIELD 也就是针对字段,同时定义 Constraint 约束条件,通过 DateValidator 类进行验证。

```
package com.ruyuan.little.project.spring.annotation;
api
                                      2
► sql
                                      3
▼ src
                                      4
                                           +import ...
  ▼ main
                                     12
    ▼ 📄 java
                                     13
      ▼ 🛅 com
                                             * @author little 校验时间是否为合法
                                     14

▼ Imruyuan

                                     15
                                             */

▼ Iittle

                                     16
                                            @Target({ElementType.FIELD}) 🚄
              m project
                                     17
                                            @Retention(RetentionPolicy.RUNTIME)
              spring
                                            @Constraint(validatedBy = DateValidator.class)
                                     18
                 annotation
                                     19
                                            public @interface IsDate {
                    @ 🖆 IsDate
                                     20
                aspect
                                                String message() default "时间格式不正确";
                                     21
                ▶ batch
                                     22
                cache
                                     23
                                                String formatter() default "";
                constants
                                     24
                controller
                                     25
                                                Class<?>[] groups() default {};
                ▶ □ dao
                                     26
                ▶ 🛅 dto
                                                Class<? extends Payload>[] payload() default {};
                                     27
                enums
                                     28
                ▶ ■ event
```

图 2 annotation 自定义标签

# 4. Aspect

Aspect 中主要用来定义 AOP 的方法,如图 3 所示,加入了环绕通知在方法执行的前后记录日志信息。

```
ControllerLogAspect around()
                                                                                                                                                                                                                                /**
* 环绕通知,在方法执行前后
                                                                                                                                                                                                                                               @param point 切入点
                                                                                                                                                                                                                                               @return 结果
@throws BusinessException
                                                                                                                                                                                                                                #
                                                                                    batch
cache
constan
controlle
dao
dto
enums
event
exception
handler
intercept
jms
mapper
service
                                                                                                                                                                                                                                                   Object target = point.getTarget();
                                                                                                                                                                                                                                                   LOGGER.info("请求处理方法:{}", target.getClass().getName() + StringPoolConstant.DOT + methodSignature.getMethod().getName());
Object[] parameterValues = point.getArgs();
/2音気冷
LOGGER.info("请求参数名:{}, 请求参数値:{}", JSONObject.toJSONString(parameterValues));
                                                                                                                                                                                                                                                 LOGGERTALE TO THE STATE OF TH
                                                                                                                                                                                                                                                                     // 执行方法
Object response = point.proceed();
                                                                               ► b service
▼ b task
ⓒ • TaskServie
► b utils
► validator
                                                                                                                                                                                                                                                 // 结束时间 long endTime = System.currentTimeMillis(); loG6ER.info("请求处理时间差:{}, 喻应结果:{}", endTime - startTime, JSON.toJSONString(response)); return response; catch (Throwable throwable) { LOG6ER.error("執行方法:{}失败, 异常信息:{}", methodSignature.getMethod().getName(), throwable); if (throwable instanceof BusinessException) { // 业务异常 return CommonResponse.fail(); }
test
target
web
gittgnore
little-project-s
m pom.xml
                                                                                                                                                                                                                                                                        ,
// 非业务异常, 封装一层异常
throw new BusinessException("方法 " + methodSignature.getMethod().getName() + " 执行失败");
```

### 图 3 aspect AOP 日志记录

#### 5 Batch

Batch 包主要包括批处理的应用,如图 4 所示,批量处理教师教学天数的信息。

```
package com.ruyuan.little.project.spring.batch;
                                   3
                                         import ...
                                  10
                                  11
                                          * @author <a href="mailto:little@163.com">little</a>
                                  12
                                            version: 1.0
                                  13
14
                                          * Description:spring实战
■ little
project
                                  15
                                         public class TeacherWriter implements ItemWriter<Teacher> {
  spring
                                  16
   annotation
                                  17
                                             @Autowired
    aspect
                                  18
                                             private TeacherMapper teacherMapper;
    batch
                                  19
       😊 🧣 StepExecutionStatsListener
                                  20
                                             @Override
       © 1 TeacherItemProcessor
                                  21 📭
                                             public void write(List<? extends Teacher> list) throws Exception {
       © TeacherRowMapper
                                  22
23
                                                  for (Teacher teacher : list){
       TeacherWriter
                                                      int update = teacherMapper.updateTeachingDays(teacher);
     cache
                                  24
   constants
                                  25
   controller
                                         }
                                  26
   ▶ □ dao
```

图 4 Batch 批处理

#### 6. Cache

Cache 包中的类用来处理缓存功能,如图 5 所示,TeacherKeyGenerator 类用来

缓存查询教师列表,根据教师类型进行查询。

图 5 cache 缓存教师列表

#### 7. Constants

Constants 包用来存放常量信息,如图 6 搜索,StringPoolConstan 类中保存了诸如: "-"、"utf-8"等常量信息。

```
package com.ruyuan.little.project.spring.constants;
                                                                                                                                                                        2
                                                                                                                                                                                                 /**
                                                                                                                                                                        3
                                                                                                                                                                        4
                                                                                                                                                                                                     * @author <a href="mailto:little@163.com">little</a>
                                                                                                                                                                        5
                                                                                                                                                                                                      * version: 1.0
va
                                                                                                                                                                        6
                                                                                                                                                                                                      * Description:string的常量池
com
                                                                                                                                                                        7
                                                                                                                                                                                                      @*/
ruyuan
                                                                                                                                                                                                  public class StringPoolConstant {
                                                                                                                                                                        8

▼ Image: Iittle Ii
                                                                                                                                                                        9
                project
                                                                                                                                                                     10

▼ Image spring

                                                                                                                                                                                                                        * -分隔符
                                                                                                                                                                     11
                                  annotation
                                                                                                                                                                                                                        ж/
                                                                                                                                                                     12
                                  aspect
                                                                                                                                                                     13
                                                                                                                                                                                                                     public static final String DASH = "-";
                                  batch
                                                                                                                                                                     14
                                  cache
                                                                                                                                                                     15

▼ lim constants

                                                                                                                                                                    16
                                                                                                                                                                                                                        * UTF-8编码
                                                © 🔓 StringPoolConstant
                                                                                                                                                                     17
                                  ▶ □ controller
                                                                                                                                                                     18
                                                                                                                                                                                                                     public static final String UTF8 = "utf-8";
                                  ▶ 🛅 dao
                                                                                                                                                                    19
                                  b dto
                                                                                                                                                                     20
                                                                                                                                                                                                                      /**
                                  ▶ ■ enums
                                                                                                                                                                                                                        * 圆点
                                                                                                                                                                     21
                                  ▶ ■ event
                                                                                                                                                                     22
                                  exception
                                                                                                                                                                                                                     public static final String DOT = ".";
                                                                                                                                                                     23
                                  handler
                                                                                                                                                                                                   }
                                                                                                                                                                     24
                                  ▶ interceptor
                                  ▶ jms
                                  mapper
                                  ▶ service
                                  ▼ 🛅 task
```

图 6 Constants 常量信息

#### 8. Controller

Controller 包中包含了后台 API 的入口,如图 7 所示,Controller 包为各个功能模块都提供了 API 的网络请求入口,在上节课中编写的 HealthController 就为例其中。

```
little-project-spring-master [little-project-spring] ~/Doc/13 - 读=
                                                            HealthController
▶ ■ .idea
                                                            package com.ruyuan.little.project.spring.controller;
▶ sql
                                                            import ...
▼ ■ src
  ▼ main
    ▼ ijava
                                                             * @author <a href="mailto:little@163.com">little</a>
       ▼ 🛅 com
                                                             * version: 1.0

▼ Imruyuan

                                                     10
                                                              * Description:健康检查的controller
                                                     11
12
13
14
15
              project
                                                            @@estController
                 ▼ 🖿 spring
                                                            public class HealthController {
                   ▶ annotation
                    aspect
                                                                 @RequestMapping(value = "/")
                   ▶ batch
                                                     16 👰
                                                                 public CommonResponse health() { return CommonResponse.success(); }
                   ▶ □ cache
                   © 🖫 StringPoolConstant
                   controller 🖿
                       © 'a ConsumerController
© 'a CouponController
                       © → HealthController
                         HelloMvcController
                       © 'a OrderController
                       © 1 TeacherController
                   ▶ lim dao
```

图 7 Controller 后台 API 的入口

#### 9、Dao

Dao 包中包含通过 JDBC 访问数据库的一些功能,如图 8 所示,JdbcBaseDao 中的 queryList 方法就是通过接受 SQL 语句和对应的参数,连接数据库进行查询,并且返回查询结果。

```
* 密码
                                                                                          39
40
  ▶ sql
                                                                                                               private String password;
                                                                                          41
42
43
44
45
      ▼ Imain
         ▼ iava
▼ iava

▼ Image: Truyuan

                                                                                                                 * 根据sql查询列表数据
                      ▼ 🛅 little
                                                                                          46
47
48

▼ Improject

                                                                                                                    <u>@param</u> sql
                             ▼ ■ spring
► ■ annotation
                                                                                                                * @param params 参数
* @param resultClazz 结果类型
* @param <>> 返回列表返现
* @return 列表结果
* @throws ClassNotFoundException
* @throws SQLException
                                                                                          49
50
51
52
53
54
55
56
57
68
61
62
63
64
65
66
67
68
                                  aspect
                                  cache
                                 ► constants
► controller
                                 ► cont
                                                                                                                * @throws InstantiationException

* @throws IllegalAccessException
                                         © № JdbcBaseDao
                                         😊 🖫 TemplateBaseDao
                                                                                                              */
public <T> List<T> queryList(String sql, Object[] params, Class<T> result
    List<T> result = new ArrayList<>();
    Connection conn = null;
    PreparedStatement preparedStatement = null;
    ResultSet resultSet = null;
                                 ▶ 🛅 dto
                                    enums
                                 ▶ ■ event
                                                                                                                     ▶ lim handler
                                 ▶ interceptor
                                     jms
                                 ▶ Imapper mapper
                                 ▼ 🛅 task
                                         C TaskService
                                                                                          69
70
71
72
73
74
75
76
77
78
79
80
                                 ▶ □ validator
          ► Iresources
                                                                                                                              resultSet = preparedStatement.executeQuery();
while (resultSet.next()) {
   T o = resultClazz.newInstance();
   Field[] fields = resultClazz.getDeclaredFields();
   for (Field field : fields) {
      ▶ test
target web
      gitignore
      alittle-project-spring.iml
                                                                                                                                             field.setAccessible(true);
Object value = resultSet.getObject(field.getName());
field.set(o, value);
       mpom.xml
► III External Libraries
                                                                                                                                      result.add(o);
                                                                                          81
```

图 8 Dao 通过 JDBC 的方式访问数据库

#### 10、DTO

用来保存对应数据库的实体类信息,如图 9 所示,dto 中的 Teacher 类就与数据库中的 Teacher 表做映射。

```
package com.ruyuan.little.project.spring.dto;
api
sal
                                                    import java.math.BigDecimal;
                                              3
src
                                             4
main
                                             5
▼ 📄 java
                                                    * @author <a href="mailto:little@163.com">little</a>
                                              6
  ▼ 🛅 com
                                                       version: 1.0
    ruyuan
                                             8
                                                     * Description: 家教老师表
      ▼ 🛅 little
                                             9
         ▼ lim project
                                                    public class Teacher {
                                             10
           ▼ lispring
                                             11
             annotation
                                             12
             aspect
                                                         * 起始页
                                             13
             ▶ batch
                                             14
                                                         */
             ▶ □ cache
                                                        private Integer startIndex = 0;
                                             15
             constants
                                             16
             controller
             ▶ 🖿 dao 👍
                                             17
                                                         * 结束页
                                             18
             ▼ 🛅 dto
                                             19
                 Comment
                                             20
                                                        private Integer endIndex = 10;
                 c 🕒 Consumer
                                             21
                 © 🔓 ConsumerCoupon
                                             22
                 Coupon
                                             23
                                                         * 开始
                 C G Order
                                                         */
                                             24
                 © № OrderDelayMessageDTO
                                             25
                                                        private Integer page = 0;
                 © 🕒 OrderMessageDTO
                                             26
                 c 🚡 Teacher
                                             27
             ▶ enums
                                             28
                                                         * 每页显示条数
             ▶ ■ event
                                             29
             exception
                                             30
                                                        private Integer size = 10;
             ▶ lim handler
                                             31
             ▶ interceptor
                                             32
                                                        /**
             ▶ 🛅 jms
                                             33
                                                         * 主键id
             ▶ ■ mapper
                                             34
                                                         */
             ▶ service
                                             35
                                                        private Integer id;
             ▼ 🛅 task
                                             36
                 □ ¹ TaskService
                                             37
                                                        /**
             ▶ 🖿 utils
                                             38
                                                         * 教书名称
             validator
                                             39
▶ ■ resources
                                             40
                                                        private String teacherName;
test 
target
```

图 9 DTO 保存对应数据库的实体类

### 11、Enums

Enums 中保存系统中的枚举信息,如图 10 所示,CommentStatusEnum 类中就保存了审核相关的枚举信息。

```
package com.ruyuan.little.project.spring.enums;
                                            2
                                            3
                                                   /**
                                                   * @author <a href="mailto:little@163.com">little</a>
                                            4
I main
                                            5
                                                    * version: 1.0
iava
                                            6
                                                    * Description:评价状态
 ▼ 🛅 com

▼ Imruyuan

                                            8
                                                  public enum CommentStatusEnum {
     ▼ 🛅 little
                                            9
        project
                                           10
                                                        * 未审核

▼ Image spring

                                           11
            ▶ ■ annotation
                                                       NON_CHECKED("1", "未审核"),
                                           12
            aspect
                                           13
                                                       /**
            ▶ batch
                                           14
                                                        * 已审核
            ▶ □ cache
                                           15
            constants
                                                       CHECKED("2", "已审核"),
                                           16
            ▶ □ controller
                                           17
            ▶ 🛅 dao
                                           18
                                                        * 未采纳
              dto
                                           19
            ▼ 🛅 enums
                                           20
                                                       NOT_ADOPTED("3", "未采纳"),
                ■ CommentStatusEnum
                                           21
                © ConsumerCouponStatusEnum
                                           22

■ CourseTypeEnum

                                           23
                                                       private String code;
                🗈 🖫 DelayMessageTypeEnum
                                                       private String name;
                🔳 🆫 EducationBusinessErrorCodeEı
                                           25
                ■ OrderStatusEnum
                                           26
                                                       CommentStatusEnum(String code, String name) {
                ■ TeacherStatusEnum
                                                           this.code = code;
this.name = name;
                                           27
                ▶ ■ event
                                           29
            exception
                                           30
            handler
                                           31
                                                       public String getCode() { return code; }
            ▶ interceptor
            ▶ Imsims
                                                       public String getName() { return name; }
                                           35
            mapper
                                           38
            service
```

图 10 Enums 枚举信息

### 12、Event

Event 包括了消息相关的信息,如图 11 所示,包括了日志消息信息 (LogonEvent)、日志消息发布(LogonEventPublisher)、日志消息监听 (LogonListener)等类。

```
package com.ruyuan.little.project.spring.event;
▶ 🖿 api
▶ sal
                                                                   import ...
                                                          6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
  ▼ main
     ▼ ijava
▼ icom
                                                                    *** <u>@author</u> <a href="mailto:little@163.com">little</a>
* version: 1.0
* Description:spring实战 定义首次登录事件
          ▼ 🛅 ruyuan
▼ 🛅 little

▼ Improject

                                                                    public class LogonEvent extends ApplicationEvent {
                       annotation
                     aspect
                                                                          * 消费者实体
                     ▶ □ cache

    ▶ constants
    ▶ controller

                                                                         private Consumer consumer;
                     ▶ 🛅 dao
                                                                         public LogonEvent(Object source, Consumer consumer) {
                                                                              super(source);
this.consumer = consumer;
                     ▶ ■ enums
                     enums
v event
C b LogonEvent
                                                           23
24
25
28
29
32
33
36
37
                          LogonEventPublisher
                                                                         public LogonEvent(@NonNull Object source) { super(source); }
                          © 1 LogonListener
                     exception
                                                                         public Consumer getConsumer() { return consumer; }
                     ► handler
► interceptor
                                                                         public void setConsumer(Consumer consumer) { this.consumer = consumer; }
                     ▶ lims
                       service
                     ▼ task

© 1 TaskService
                     ▶ ■ utils
     resources
```

### 图 11 Event 消息相关信息

### 13、Exception

Exception 中包括了异常信息,如图 12 所示,BusinessException 继承了

RuntimeException 类,如果有需要可以在 RuntimeException 的基础上定制化自己

的异常类。

```
■ little-project-spring-master [little-project-spring] ~/Doc/13 - 读书章
                                                            BusinessException
▶ idea .idea
                                                            package com.ruyuan.little.project.spring.exception;
  sal
                                                             * FEBS系统内部异常
  ▼ main
     ▼ ijava
                                                               @author little
       ▼ 🛅 com
          ▼ 🛅 ruyuan
                                                            public class BusinessException extends RuntimeException {
            ▼ 🛅 little

▼ Improject

                                                                private static final long serialVersionUID = -994962710559017255L;
                 ▼ 🛅 spring
                                                    11
12
15
16
                     annotation
                                                                public BusinessException(String message) { super(message); }
                   ▶ ■ aspect
                   ▶ 🛅 batch
                   cache
                   constants
                   controller
                   ▶ □ dao
                   ▶ 🛅 dto
                   ▶ ■ enums
                     event

▼ mexception

                       BusinessException

NoRollbackException
```

图 12 Exception 异常信息

# 14、Handler

上面提到了 Exception 是用来定义异常信息的,那么这里的 Handler 是用来处理异常信息的。如图 13 所示,这里定义了 GlobalExceptionHandler 来处理全局异常。

```
▼ Iittle-project-spring-master [little-project-spring] ~/Doc/13 - 读书
                                                                                             GlobalExceptionHandler
 > lapi
> api
> sql
                                                                                              package com.ruyuan.little.project.spring.handler;
                                                                                            import ...
      ▼ main
                                                                                              * <u>@author</u> little
                                                                                             @ControllerAdvice
                                                                                             @ResponseBody
@Order(value = Ordered.HIGHEST_PRECEDENCE)
public class GlobalExceptionHandler {
                        project
project
project
project
project
project
project
project
project
                                                                                                      * 日志组件
                                 cache
                                  constants
                                                                                                    private static final Logger LOGGER = LoggerFactory.getLogger(GlobalExceptionHandler.class);
                                  controller
dao
                                                                                                    @ExceptionHandler(value = Exception.class)
public CommonResponse handleException(Exception e) {
    LOGGER.error("系统异常错误信息:{}", e);
    return CommonResponse.fail();
                               ▶ menums
                                 event
                                                                                                    @ExceptionHandler(value = BusinessException.class)
public CommonResponse handleBusinessException(BusinessException e) {
    LOGGER.error("系統异常错误信息:{}", e);
    return CommonResponse.fail();
                               interceptor
                                  ims jms
                               mapper
mapper
task
TaskService
```

图 13 Handler 处理异常

# 15 Interceptor

字节码拦截器,Java 文件在编译成 Class 之后才能在 JVM 中运行,编译成的 Class 文件被称为字节码文件。字节码拦截器就是在这个层面对代码进行拦截,在 执行的 action 的前后嵌入需要执行的代码也是 AOP 的一种执行方式。如图 14 所示,这里定义了 CharacterEncodingInterceptor 对登陆请求进行拦截,拦截需要登陆的接口是否传入手机号。

```
CharacterEncodingInterceptor
ideaiapisqlsrc
                                                                      package com.ruyuan.little.project.spring.interceptor;
   qı src main
v işava
v ruyuan
v iştle
v işpriject
v işpriject
                                                             11
12
13
14
15
16
17
18
19
20
21
22
21
23
24
25
26
27
28
29
                                                                       ***

* <u>@author</u> <a href="mailto:little@163.com">little</a>

* version: 1.0

* Description:字符编码拦截器
                                                                      public class CharacterEncodingInterceptor implements HandlerInterceptor {

▼ Image spring

                      ▶ annotation
                                                                            private static final Logger LOGGER = LoggerFactory.getLogger(CharacterEncodingInterceptor.class);
                      baspect
batch
cache
constants
controller
                                                                                request.setCharacterEncoding(StringPoolConstant.UTF8); response.setCharacterEncoding(StringPoolConstant.UTF8); response.setCharacterEncoding(StringPoolConstant.UTF8); LOGGER.info("LoginInterceptor.preHandle()"); return false;
                                                                            public boolean preHandle(HttpServletReguest reguest, HttpServletResponse response,
                        🛅 dao
                      b dto
                       ▶ ms jms
                        mapper mapper
                        service
                      task
TaskService
                                                                            public void afterCompletion(HttpServletRequest request, HttpServletResponse response,
                                                             38
39
40
                                                                                 Object handler, Exception ex)
LOGGER.info("LoginInterceptor.afterCompletion()");
     test
► limitarget
```

图 14 Interceptor 字节码拦截器

### 16、JMS

主要用来定义 JMS 消息队列的配置和监听器,如图 15 所示,

OrderDelayMessageListener 就定义延迟消息的监听器。

```
OrderDelayMessageListener
api
sql
src
w main
w ipva
com
w ittle
w iproject
w ittle
w iproject
w ispring
b annotation
b aspect
b batch
constan
                                                                                          package com.ruyuan.little.project.spring.jms;
                                                                                            ***
** <u>@author</u> <a href="mailto:little@163.com">little</a>
* version: 1.0
* Description:延时消息listener组件
                                                                                          public class OrderDelayMessageListener implements MessageListenerOrderly {
                                                                                                   private static final Logger LOGGER = LoggerFactory.getLogger(OrderDelayMessageListener.class);
                                constants
controller
dao
dto
enums
event
                                                                                                   * 优惠券service组件
                                                                                                  private OrderService orderService;
                                exception handler
                                interceptor
ims
                                                                                                  public ConsumeOrderlyStatus consumeMessage(List<MessageExt> list, ConsumeOrderlyContext consumeOrderly
for (MessageExt ext : list) {
    String content = new String(ext.getBody(), StandardCharsets.UTF_8);
    LOGGER.info("received order delay message:{}", content);
                                     © % MgConsumerConfiguration
                                     © = MqDelayConsumerConfiguration
                             // 订单消息
OrderDelayMessageDTO orderMessage = JSON.parseObject(content, OrderDelayMessageDTO.class);
                                                                                                               // 订单内容
Order order = JSON.parseObject(orderMessage.getContent(), Order.class);
DelayMessageTypeEnum delayMessageType = orderMessage.getDelayMessageType();
                                                                                                               try {
    if (DelayMessageTypeEnum.PAY_DELAY.equals(delayMessageType)) {
        // 订单取消
        orderService.cancelOrder(order.getId());
    } else if (DelayMessageTypeEnum.FINISH_DELAY.equals(delayMessageType)) {
        // 订单完成
        orderService.finishOrder(order.getId());
    }
                             ▶ <u>utils</u>
                             ▶ ■ validator
        ▶ ■ resources
     target web
```

#### 图 15 JMS 定义消息的配置和监听器

### 17、Mapper

主要实现数据操作模块,如图 16 所示,这里对 order 订单进行了,add、

findByld 等一系列操作。

```
* <u>@author</u> <a href="mailto:little@163.com">little</a>
* version: 1.0
* Description:订单mapper组件
                                                                                                                                                                                                                                                                     public interface OrderMapper {
                                                                                 yuan
little
project
list spring
list annotation
list aspect
list aspect
list acache
list constants
list controller
list dao
                                                                                                                                                                                                                                                                                                                                                <u>@param</u> order 订单信息
<u>@return</u> 结果雷暴
                                                                                                                                                                                                                                                                                                                                 List<Order> getPage(Order order);
                                                                                       b constants
b constants
capable
date
b date

                                                                                                                                                                                                                                                                                                                                          * 根据id获取订单详情
                                                                                                                                                                                                                                                                                                                                       * @param id 订单id
* @return 订单详情
                                                                                                                                                                                                                                                                                                                                 Order findById(Integer id);
                                                                                                                                                                                                                                                                                                                               /**
* 下订单
                                                                                                                                                                                                                                                                                                                                     * @param order 订单信息
* @return 操作结果
                                                                                                                                                                                                                                                                                                                                  int add(Order order);
                                                                                                                                                                                                                                                                                                                               /**
* 根据状态和是否已统计查询订单信息
                                                                                                                                                                                                                                                                                                                                        ► Ingresources
► Ingresources
► Ingresources
► Ingresources
                                                                                                                                                                                                                                                                                                                                 List<Order> findByOrderStatusAndCounted(@Param("status") String status, @Param("counted") String counted);
```

图 16 Mapper 数据操作

### 18 Service

主要负责对业务服务的处理,直接会被 Controller 层调用,大部分的业务代码会在这里完成。一般而言在 Service 中都会定义一个接口和对应的实现。如图 17 所示,定义了 ConsumerService 的接口,接口中定义需要实现的方法,然后在 ConsumerServiceImpl 中定义对应的实现。

```
import ...
src
main
                                                             9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
125
26
27
28
29
  ▼ ijava
▼ imcom
                                                                        **<u>@author</u> <a href="mailto:little@163.com">little</a>
* version: 1.0
* Description:消费者service组件
         ▼ 🛅 ruyuan
            ▼ 🛅 little

▼ Image spring

                                                                       public interface ConsumerService {
                     aspect
                     ▶ 🖿 batch
                                                                             /**
* 添加消费者
                    ► 🛅 cache
                    ▶ ☐ constants
                    ▶ ☐ controller
                                                                              * <u>@param</u> consumer 消费者实体
* <u>@return</u> 结果
                    ▶ 🛅 dao
                    ▶ 🖿 dto
▶ 🖿 enums
                                                                             int add(Consumer consumer);
                     ▶ 🖿 event
                    exceptionhandler
                                                                              * 根据条件查询总条数
                    ▶ interceptor
                    ▶ 🛅 jms
                                                                              * <u>@param</u> consumer 消费者id
                    ▶ 🛅 mapper
                                                             30
31
                                                                              * @return 结果
                     ▼ 🖿 service
                                                             31
32 •↓
33
34
35
36
37
38
                                                                             int getPageTotal(Consumer consumer);
                             ConsumerServiceImpl
                              © 🖫 CouponServiceImpl
                                                                             /**
* 分页查询消费者
                             © % OrderEvenManagerImpl
                             © • OrderServiceImpl
                             © № TeacherServiceImpl
                                                                              * @param consumer 查询条件
                          1 & ConsumerService
                                                                              * @return 结果

    CouponService
    GrderEventManager

                                                             39
40 Q
                                                                             List<Consumer> getPage(Consumer consumer);

    OrderService
    TeacherService

                                                             41
42
43
44
                                                                              * 根据id查询消费者详情
                    ▶ 🖿 utils
                    ▶ 🖿 validator
```

图 17 Service

#### 19、Task

用来执行系统中的计划任务,可以和 Batch 配合完成定时更新和批量处理的功能。如图 18 所示,TaskService 中就有处理教师教学天数更新(updateTeachingDays)的任务。

```
little-project-spring-master [little-project-spring] ~/Doc/13 - 读书筆
                                                                                TaskService
                                                                                package com.ruyuan.little.project.spring.task;
► api
► sql
▼ src
  sql src main
v java
v com
v truyuan
v titt
                                                                                 ** <u>@author</u> <a href="mailto:little@163.com">little</a>
* version: 1.0
* Description:spring实战
                                                                      ▼ little
▼ liproject
                                                                               public class | TaskService {
    Logger | Log | LoggerFactory.getLogger(getClass());

▼ lim spring

                        ➤ annotation
➤ aspect
➤ batch
                                                                                      private TeacherService teacherService;
                         ▶ □ cache
                        ▶ □ constants▶ □ controller
                                                                                      private OrderService orderService;
                         ▶ lim dao
                        ▶ 🖿 dto
                        enums
event
                                                                                     private JobParametersBuilder jobParametersBuilder;
                         exception
                                                                                     @Autowired
private Job teacherJob;
                        ▶ lim handler
                        ▶ interceptor
▶ ims
                                                                                     @Autowired
                        ► Imappo

► Imappo

► service
                                                                                     private JobLauncher jobLauncher;
                        ▼ task

C 1 TaskService
                                                                                       * 定时任务更新教师
* 此处更新教师信息,将teacher缓存全部清除
                        ▶ □ utils
                        ▶ Nalidator
                                                                                      @CacheEvict(value = "teacher", allEntries = true)
     test
                                                                                      public void updateTeachingDays() throws Exception {
▶ limitarget
▶ web

ightignore
                                                                                           /*
* Spring Batch Job同一个job instance, 成功执行后是不允许重新执行的【失败后是否允许重跑,
* 可通过配置Job的restartable参数来控制,默认是true】,如果需要重新执行,可以变通处理,
* 添加一个JobParameters构建类,以当前时间作为参数,保证其他参数相同的情况下却是不同的job instance
   alittle-project-spring.iml
m pom.xml
                                                                                           */
log.info("开始更新教师教学天数和评分等信息");
jobParametersBuilder.addDate("date", new Date());
jobLauncher.run(teacherJob, jobParametersBuilder.toJobParameters());
log.info("更新教师教学天数和评分等信息完成");
                                                                                               StopWatch sw = new StopWatch();
sw.start();
```

图 18 计划任务

#### 20 Utils

作为工具箱属于系统的通用模块,如图 19 所示,DateUtils 中保存大量处理日期的

功能函数。

```
▼ IIIalittle-project-spring-master [little-project-spring] ~/Doc/13 - 读书单
                                                                                                                                                                                                                                                                                                                                                                                                                             DateUtils
               idea
                                                                                                                                                                                                                                                                                                                                                                                                                                 package com.ruyuan.little.project.spring.utils;
         api sql
sscl
ssc
w main
w java
w mittle
w moriyuan
w
                                                                                                                                                                                                                                                                                                                                                                                                                             import ...
                                                                                                                                                                                                                                                                                                                                                                        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
44
44
45
46
47
                                                                                                                                                                                                                                                                                                                                                                                                                                     /**
* <u>@author</u> <a href="mailto:little@163.com">little</a>
* version: 1.0
* Description:spring实战
                                                                                                                                                                                                                                                                                                                                                                                                                             public class DateUtils {
                                                                                                                                                                                                                                                                                                                                                                                                                                                                     * 日志管理组件
                                                                                                                                                                                                                                                                                                                                                                                                                                                             private static final Logger LOGGER = LoggerFactory.getLogger(DateUtils.class);
                                                                                                                                                                                                                                                                                                                                                                                                                                                             public static final String FULL_TIME_SPLIT_PATTERN = "yyyy-MM-dd HH:mm:ss";
                                                                                                                                               ▶ 🛅 dao
                                                                                                                                           dao
dto
dto
dto
december decem
                                                                                                                                                                                                                                                                                                                                                                                                                                                             public static final String Y_M_D = "yyyy-MM-dd";
public static final int Y_M_D_LENGTH = 10;
public static final int FULL_TIME_LENGTH = 19;
public static final String Y_M = "yyyy-MM";
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  计算两个时间相隔年份
                                                                                                                                                                                                                                                                                                                                                                                                                                                                     * @param startDate 开始时间
* @param endDate 结束时间
* @return 结果
                                                                                                                                                       task utils
                                                                                                                                                                      © 1 DateUtils
                                                                                                                                                                                                                                                                                                                                                                                                                                                             #/
public static int betweenOfYear(Object startDate,Object endDate) {
   if (startDate == null || endDate == null){
        throw new BusinessException(EducationBusinessErrorCodeEnum.PARAMETER_NOT_EMPTY.getMsg());
}
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           }
LocalDate startLocalDate = toLocalDate(startDate,null);
LocalDate endLocalDate = toLocalDate(endDate,null);
return Period.between(startLocalDate,endLocalDate).getYears();
```

图 19 Utils 工具箱

#### 21 Validator

Validator 起到验证的作用,如图 20 所示,这里的 DateValidator 类对日期类型进行了验证,和前面介绍的 Annotation 包配合使用。

```
DateValidator
                                                                                                                                                                                                                                                   package com.ruyuan.little.project.spring.validator;
                                                                                                                                                                                                                    13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
21
33
34
35
34
35
36
37
                                                                                                                                                                                                                                                                  校验是否为合法的 时间格式
                                                                                                                                                                                                                                                     public class DateValidator implements ConstraintValidator<IsDate, String> {
                                                                                                                                                                                                                                                                      /**
* 日志组件
                                                                                                                                                                                                                                                                      private static final Logger LOGGER = LoggerFactory.getLogger(DateValidator.class);
                                                                                                                                                                                                                                                                       public void initialize(IsDate isDate) {
                                                                                                                                                                                                                                                                     ► lim handler
                                                                                 ▶ interceptor
    ► Ingresources
► Intest
► Intest
► Intest
► Intest
► Intest
► Intest
■ Intert
■ Intest
■ Int
                alittle-project-spring.iml
                 m pom.xml
```

图 20 Validator 验证

### 22、总结

今天把互联网教育系统的服务包给大家介绍了一下,包括模块是如何划分的,以及 每个模块都负责哪些功能和信息的处理。下节课,会对儒猿实战平台做一个介绍, 下期见,拜拜。