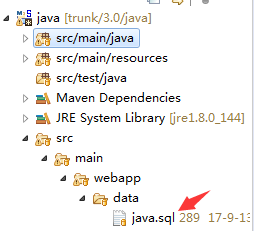
# 第一部分：后台开发

## 1.导入框架基础表

在项目webapp的data文件夹中有个sql文件



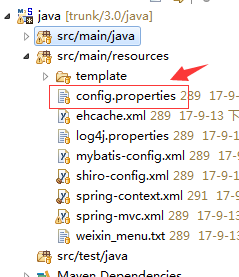
把它导入自己新项目的数据库中，框架系统表前缀是sys\_，所以项目的业务表前缀不可以设置为sys\_。

分别是sys\_province（省表）、sys\_city（市表）、sys\_district（区表）、sys\_user（管理员表）、sys\_role（角色表）、sys\_perm（权限表）、sys\_role\_perm（角色-权限表）、sys\_role\_user（角色-用户表）、sys\_menu（系统菜单表）、sys\_dict（字典表），这些表都是框架表，原则上不能修改其结构，需要时根据项目需求自行修改。

另外还有两个测试表提供代码生成测试。

## 2.配置并启动

在resources资源文件夹下有个config.properties文件

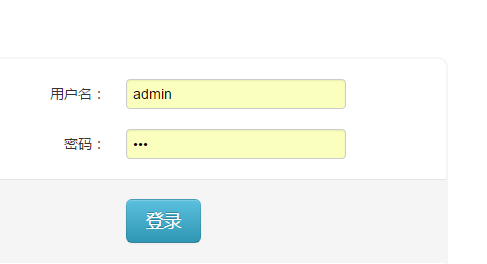


jdbc的连接自己修改；其中project.name是项目工程名，导入框架后，自己修改工程名，默认是java，project.name需改为与工程名一致；workplace是eclipse工程的工作目录，填错会导致不能生成代码，建议鼠标右击工程——properties，在Resourcede的Location那里复制一下。

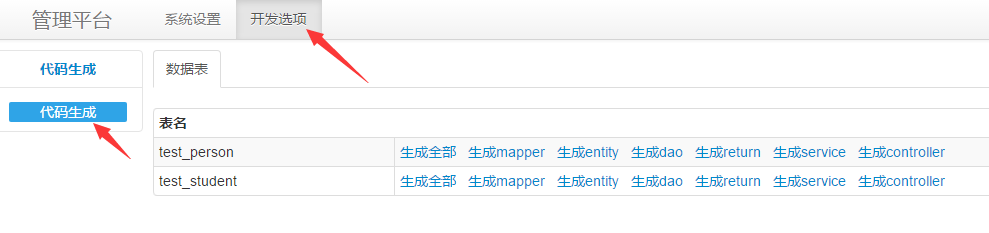
启动tomcat。

## 3.代码生成

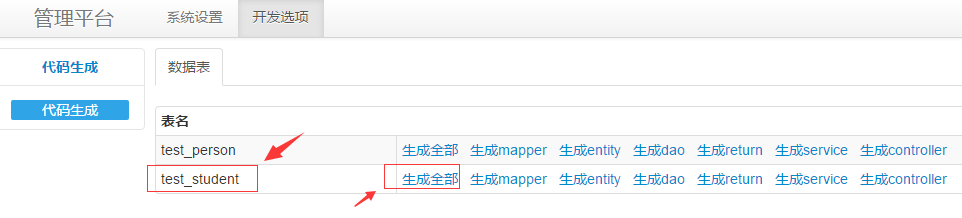
tomcat启动好之后，输入http://localhost:8080/java/sys/



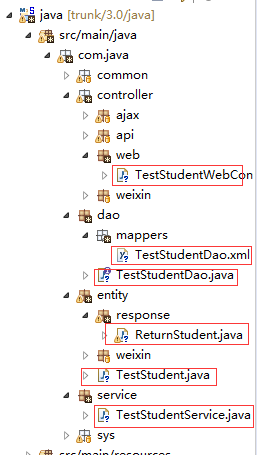
默认账号密码是admin 123，登录进去后点开发选项的代码生成



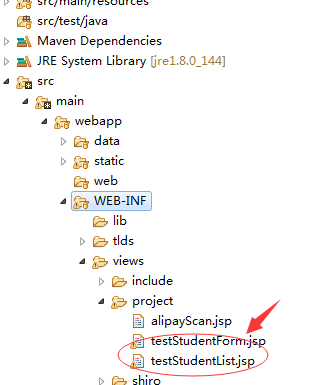
这里以表test\_student为例，点击生成全部



然后鼠标点击工程，按F5刷新一下，可以看到多出了一些java类



还有两个jsp



这样controller、service、dao、mapper.xml、实体类、返回实体类、列表jsp、表单jsp都生成好了，再往上面做开发即可。

注意：修改后如果再点击生成，会把原有的java类覆盖掉，请做好SVN、git等保存

## 4.菜单创建以及权限设置

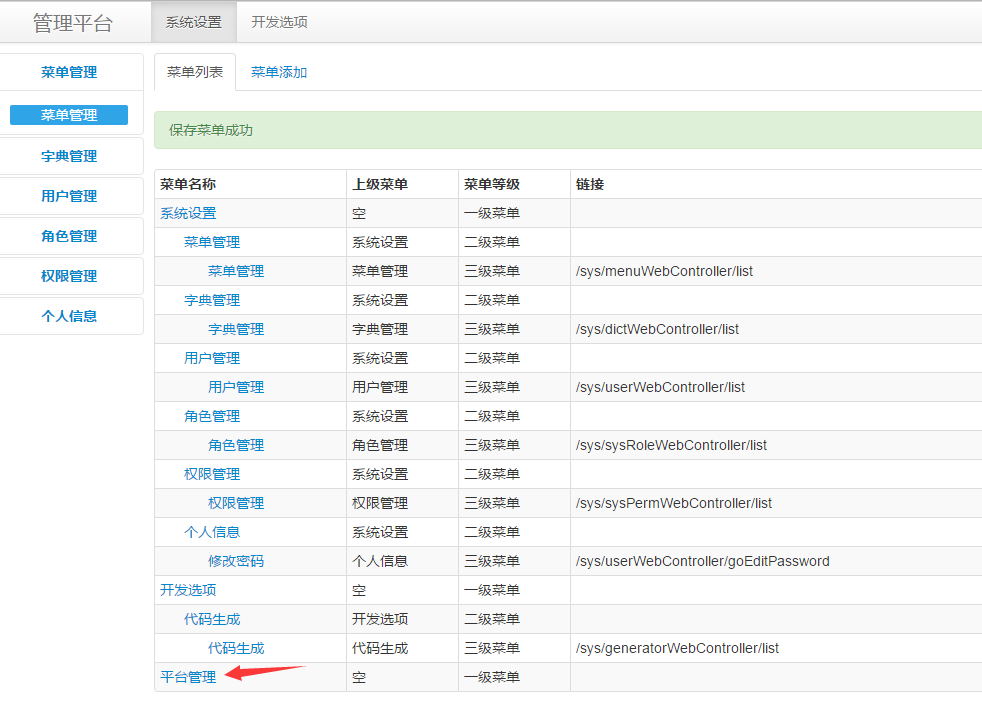
重启tomcat，重新登录。



点击系统设置——菜单管理——菜单添加



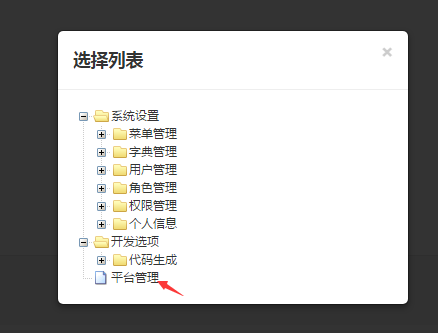
先添加一个一级菜单，所以上级菜单不选，链接留空不用写，排序随意，保存，就会出现在这里



然后再点击菜单添加，添加一个二级菜单，取名为“学生管理”

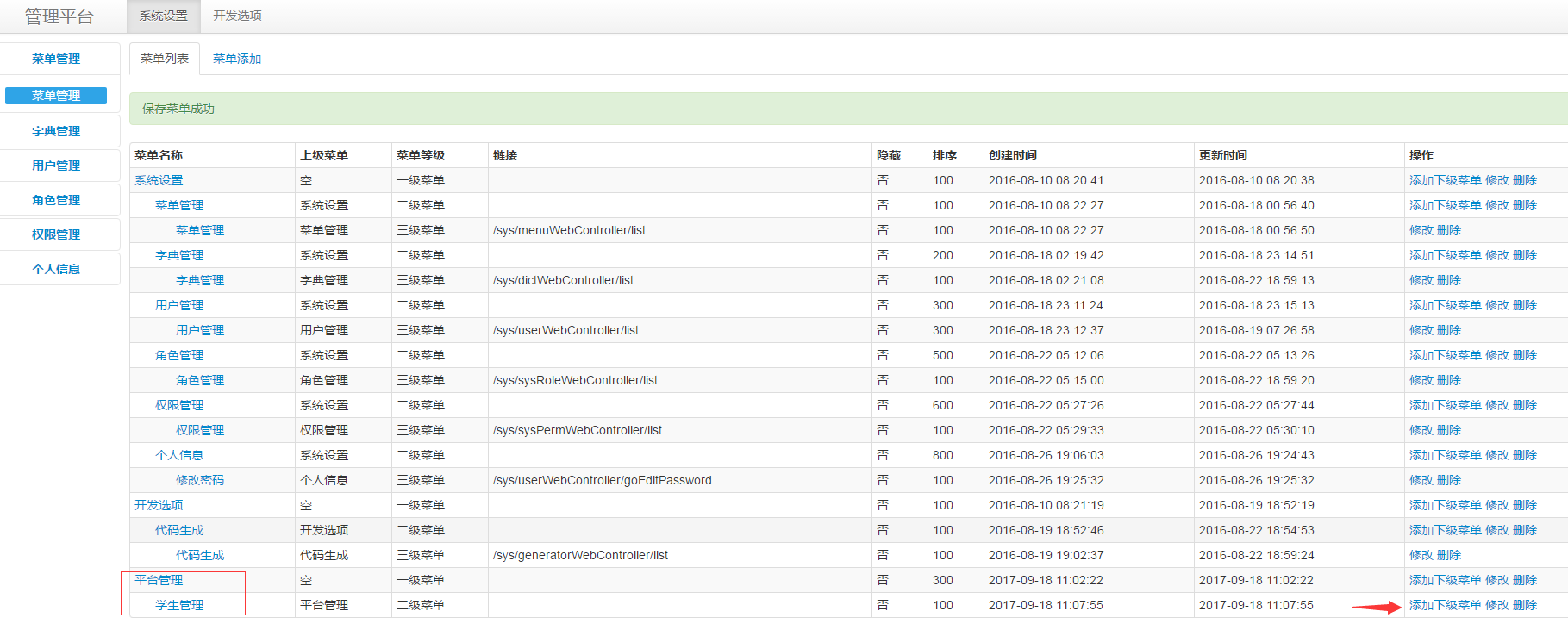


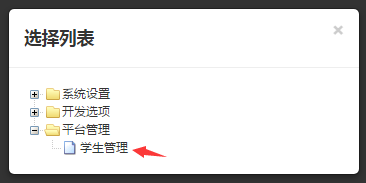
这时候我们把它的上级菜单选为刚刚添加的“平台管理”，鼠标双击，保存



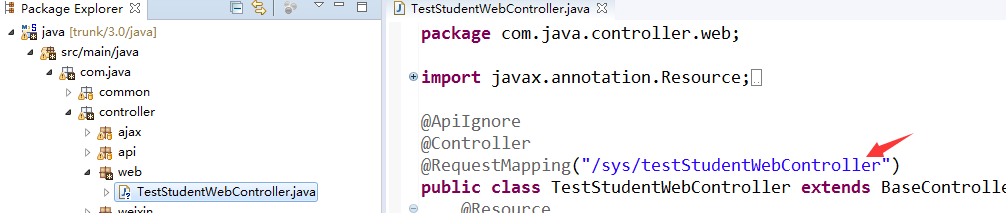


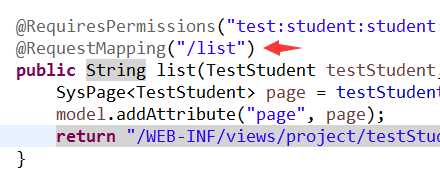
然后再添加一个三级菜单，名字我也叫做“学生管理”





链接值填/sys/testStudentWebController/list，去生成的控制器copy

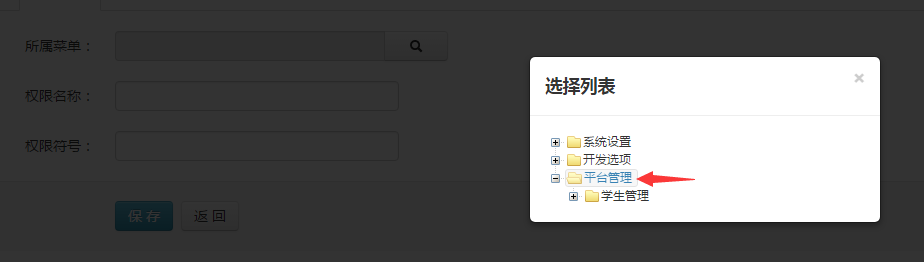




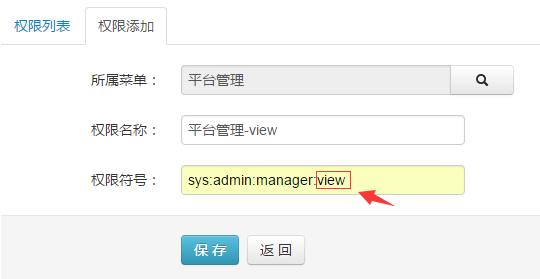
保存，这时候，我们已经添加一个一级菜单、二级菜单、三级菜单，但是刷新页面还是没有出来，因为没有添加菜单对应的一些权限。

点击权限管理——权限添加，所属菜单选中“管理平台”





双击，



权限名称随便改，权限符号不能，必须要有view，shiro权限符号有相应规范，如view是查看，edit是编辑和添加，delete是删除。

“:”符号的作用：

如aa:bb:\*和aa:bb:cc，权限aa:bb:\*可以访问aa:bb:cc，而权限aa:bb:cc不能访问aa:bb:\*，

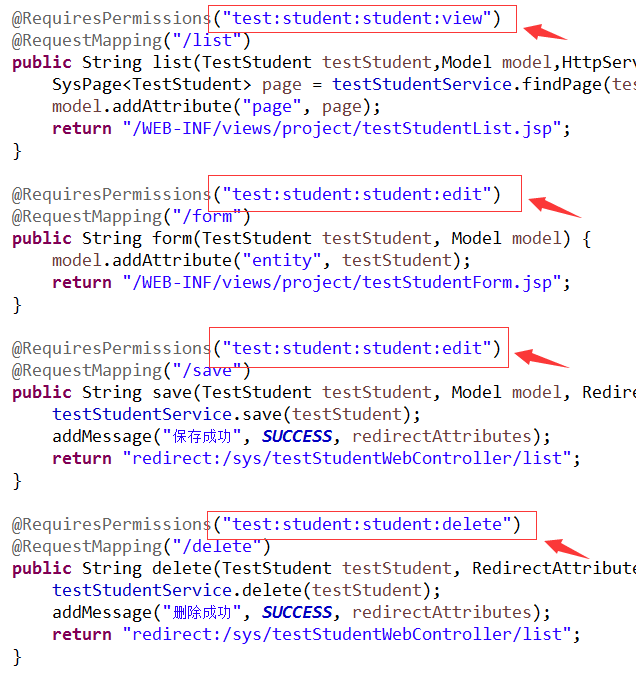
“:”相当于一个权限层次分隔符，至于分多少层，随意，一般取名为->

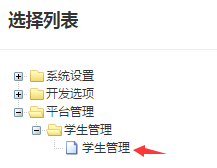
“项目:模块名:表名:view/edit/delete”

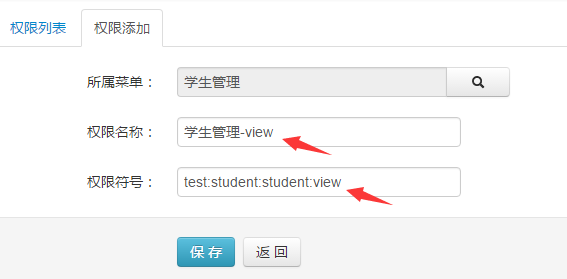
这时候一级菜单“平台管理”的view权限就添加好了

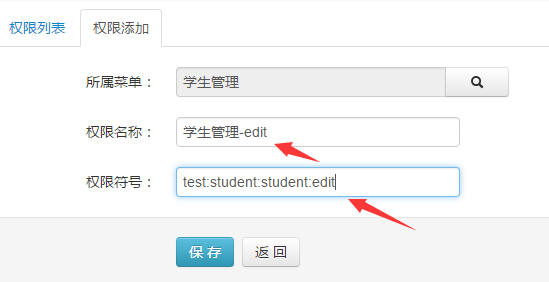
再添加test\_student表的view、edit、delete权限。

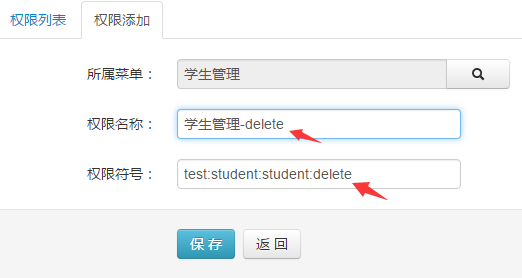
进入student的控制器，copy里面的权限符号（已自动生成好）



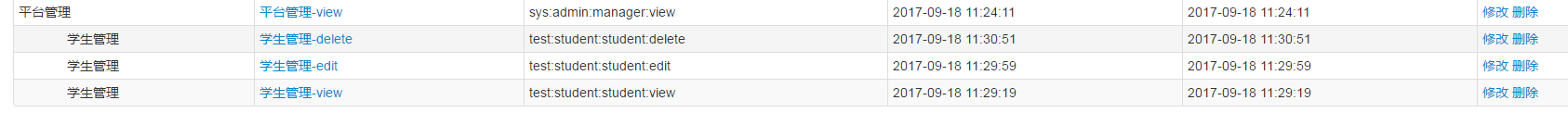




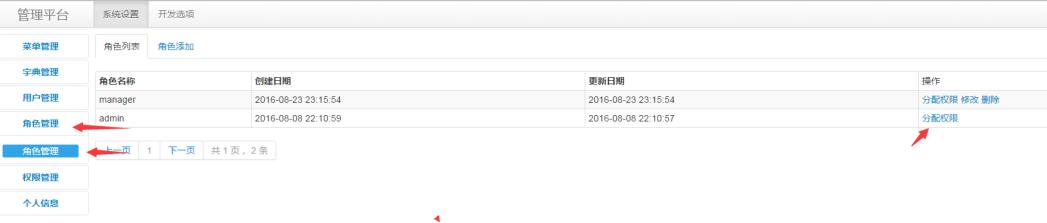


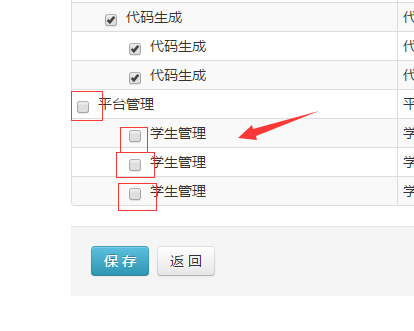


三个权限添加好，出现在这里

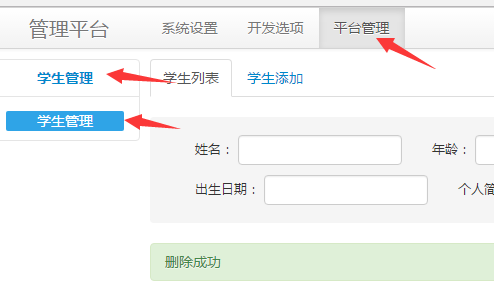


然后点击角色管理，当前角色分配权限

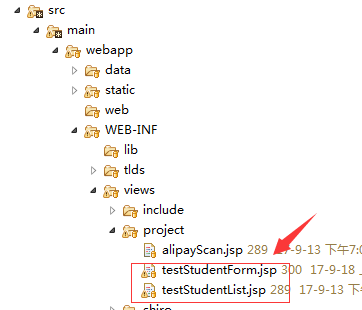




保存，刷新，就出来了



这是候基本的增删查改已经做好，可以自己添加一些数据试试，jsp代码已经在刚刚点击代码生成的时候生成好，来自于这两个jsp文件



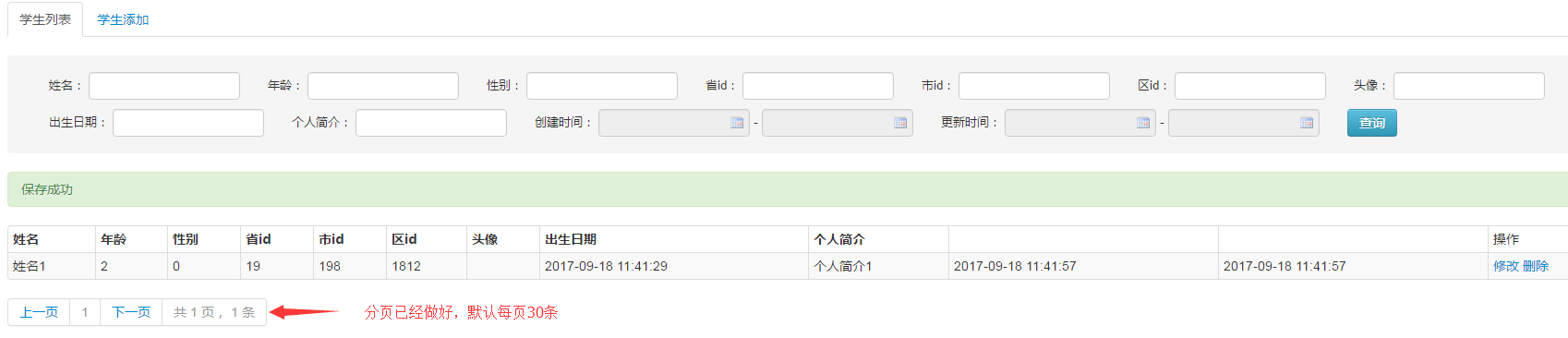
这时，如果表中有字段的名字为province\_id（省id）、city\_id（市id）、district\_id（区id）的话，往往需要做省市区三级下拉选择联动，已经自动生成好



前提是表的字段名取名为为province\_id、city\_id、district\_id，



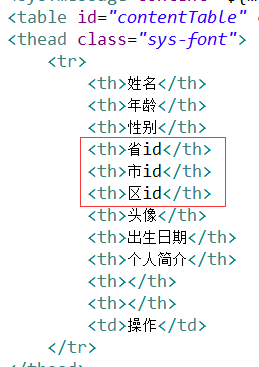
保存，可以看到查询列表中的数据



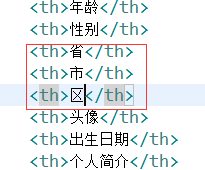
## 5.框架JSP组件的使用

### 5.1省市区转换表达式

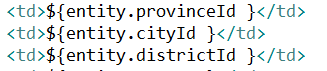
可以看到，sys\_province、sys\_city、sys\_district表中有省市区名称、省市区id，而其他表保存数据一般是保存省市区的id，这时候展示名称就要在列表中for循环根据student数据中的省市区id把省市区的名称查出来再展示。打开testStudentList.jsp，



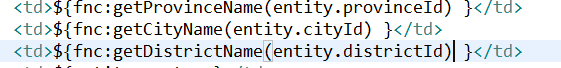
改为



把



改为



然后再打开studen列表



变成这样



### 5.2字典表达式

例如性别字段存储0和1，0是女，1是男，或者一些业务需要有审核状态字段，-1是指不通过，0是未提交，1是处理中，2是已通过等等。利用框架字典表可以自动转化对应的中文意思。

点击字典管理，字典添加





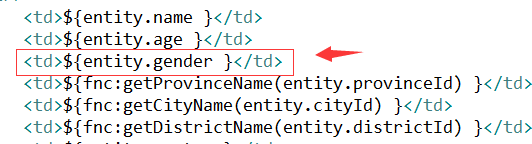
保存，再添加



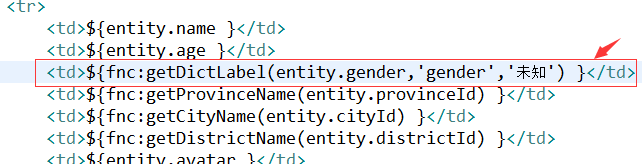
这时候就添加好了性别字典



再打开testStudentList.jsp，把



改为



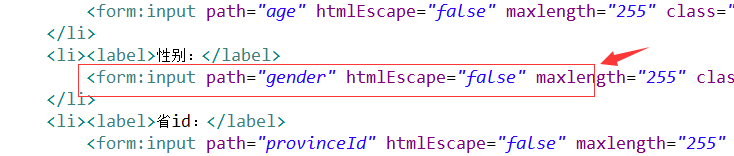
getDictLabel表达式中第1个参数是需要转化的值，第2个参数是字典类型，第3个是不匹配或者为空的时候显示的值。保存，再看看列表，变成这样



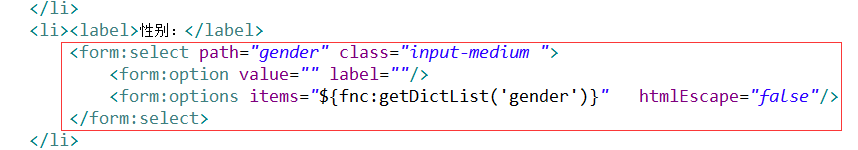
但是查询的框还是这样的



我们把



改为



就会变成这样

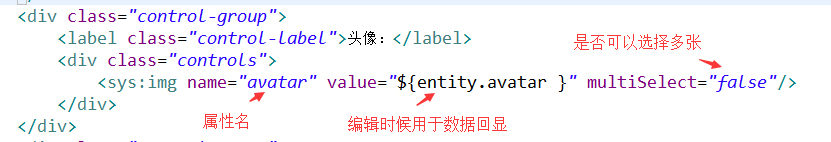


### 5.3图片选择器

把



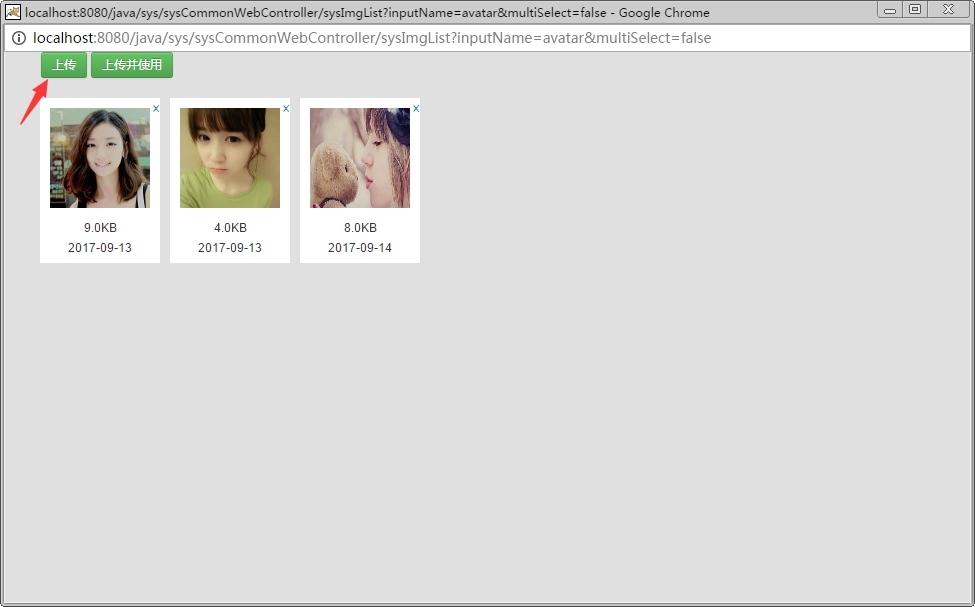
改为



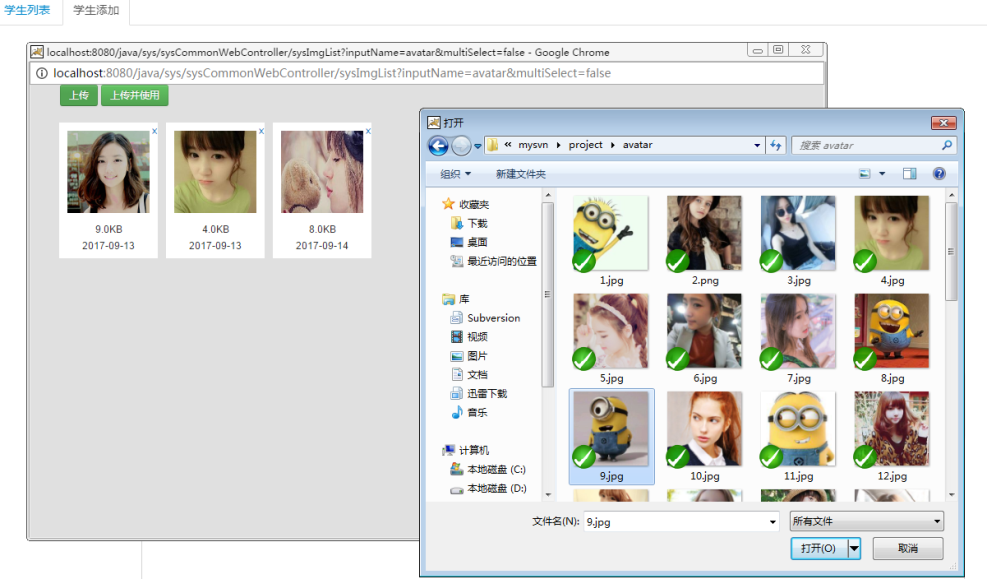
编辑表单就会变成这样



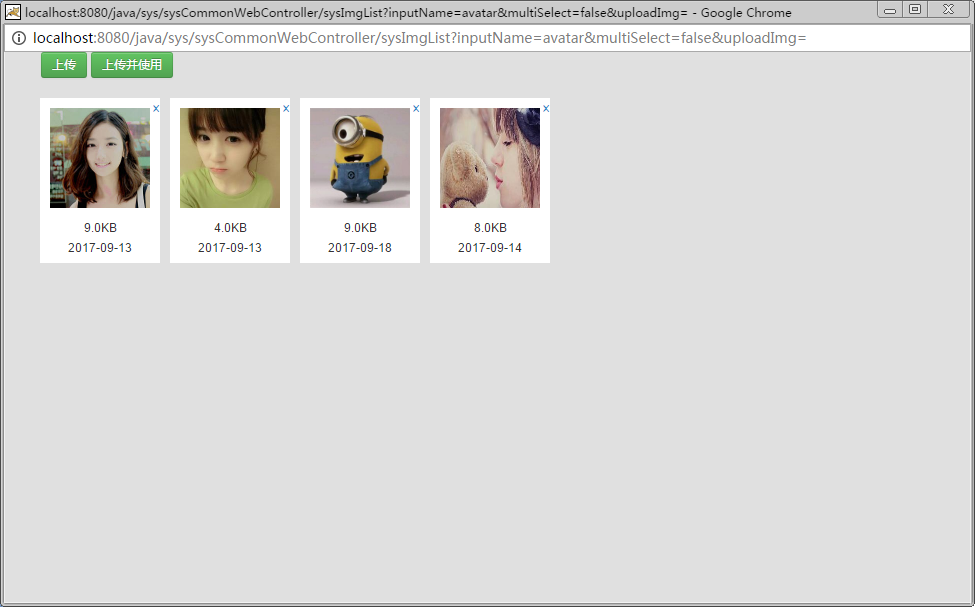
点中会弹出图片选择器，点击上传可以添加图片



选中要上传的图片



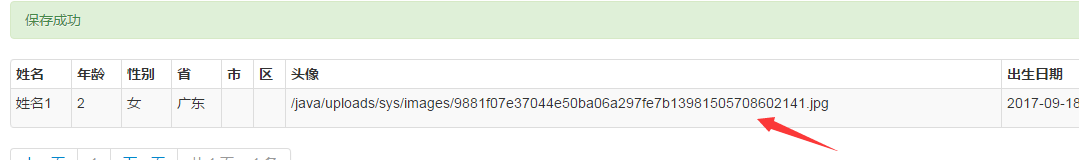
就可以上传图片



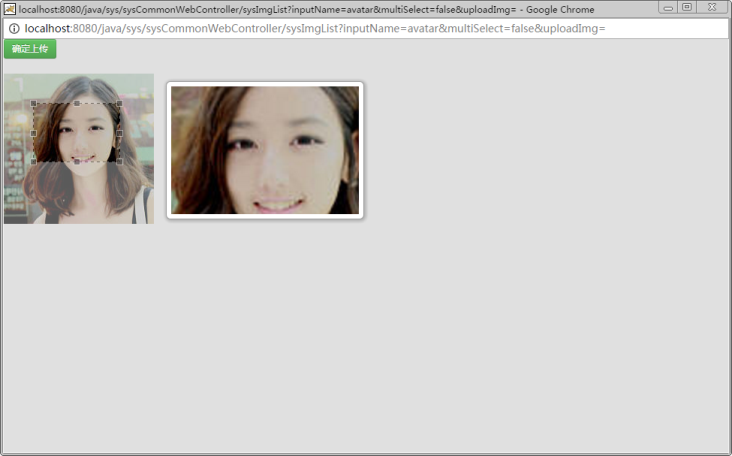
双击图片，回到表单



保存，图片的服务器路径会赋值到avatar字段中



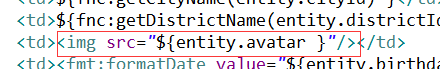
如果点的是“上传并使用”，会多出来一个截图功能，矩形宽高比可以自行去改



在列表.jsp中把



改为

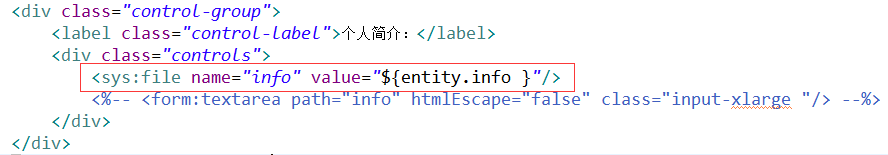


即可

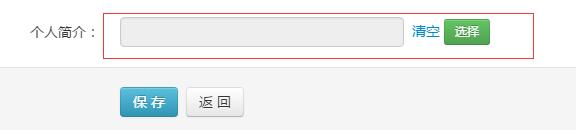


### 5.4文件选择器

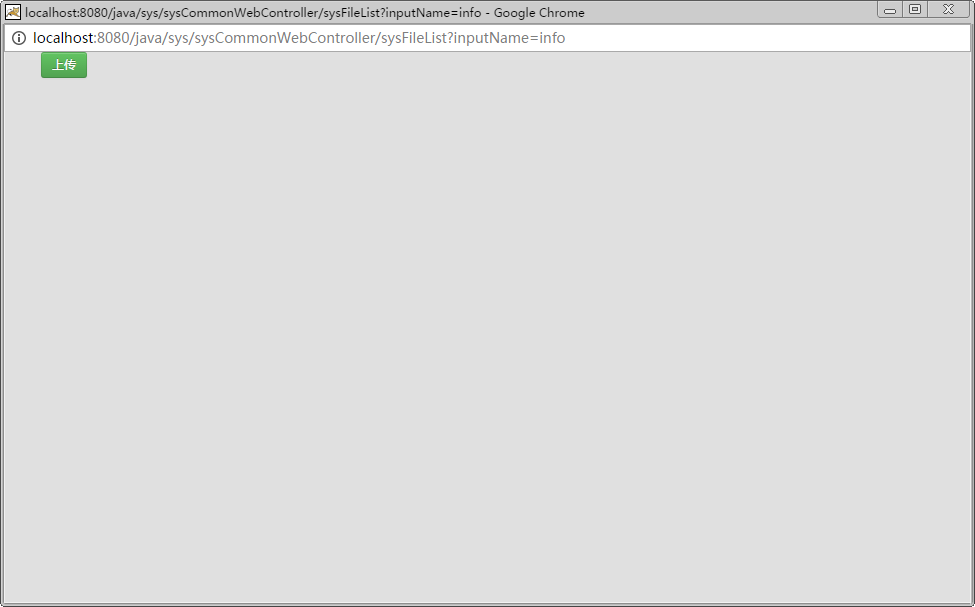
文件选择器与图片类似，如：



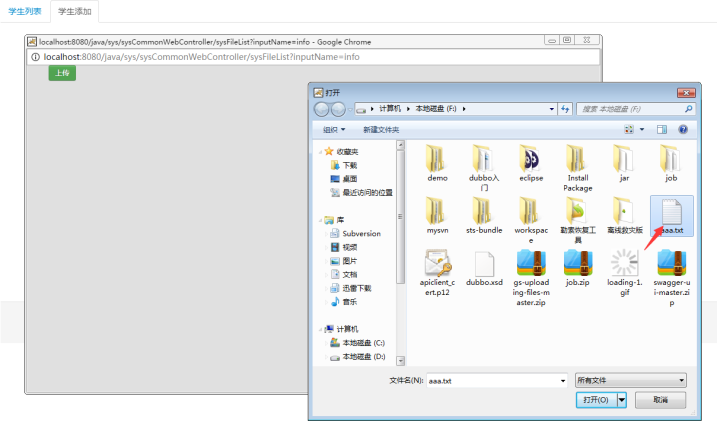
编辑表单就会变成这样：



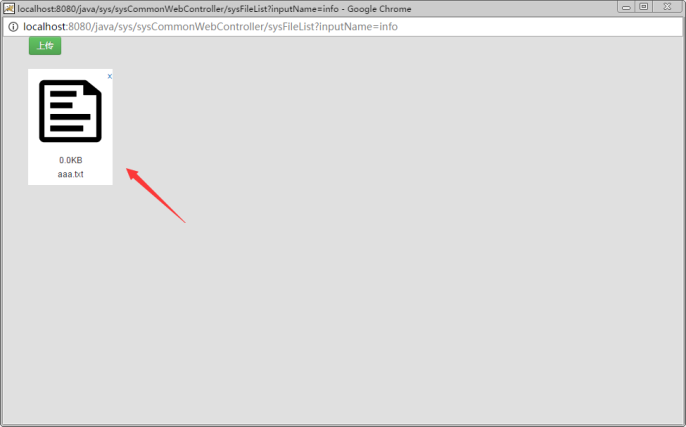
点击选择



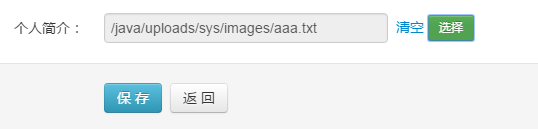
点击上传



上传文件



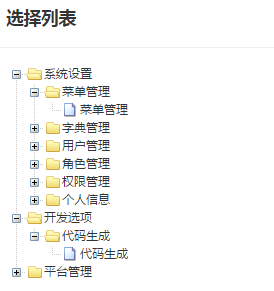
双击上传的文件，会自动把文件的服务器路径赋值到字段的表单值里面



在列表中把字段值显示成<a>标签的href值，点击就可以下载

### 5.5树形结构选择器

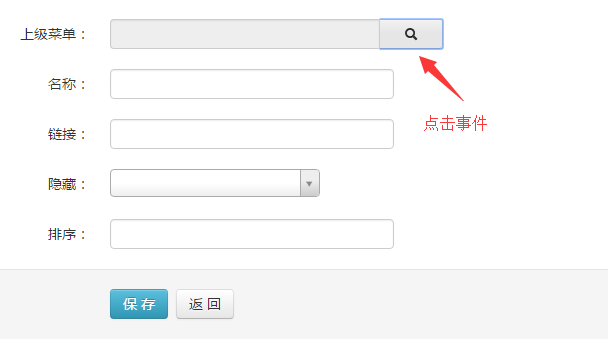
就像这种



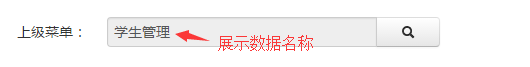
如果一些字段的值（如果商品的分类，分来有很多级），是来源于这种树形的数据结构，传统的做法：

1.后台部分：写一个接口，把一级数据查询出来，再把二级、三级、xxx级数据查出来。

2.前端引用一个树形js控件，根据逻辑添加点击事件



再写一个双击事件，把鼠标双击的id值（如分类id）返回表单隐藏的input中，把id对应的名称（如分类名称）显示在展示input中





现在只需要第一步，写接口，把数据赋值给类SysTree返回，具体参考

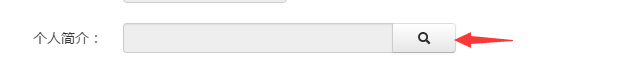
TestApiController这个类的getTreeList方法

2.给表单jsp中写

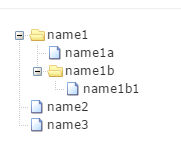
<sys:tree url=*"接口地址"* name=*"属性名"* value=*"*${entity.属性名 }*"* required=*"true/false"*/>



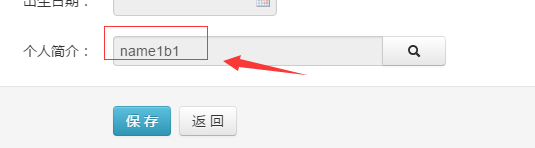
表单jsp中对应变成这样



点击就会出来数据



双击可以赋值

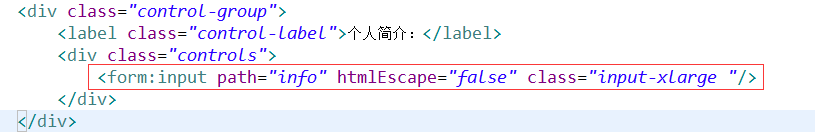


## 6.UEditor使用和配置

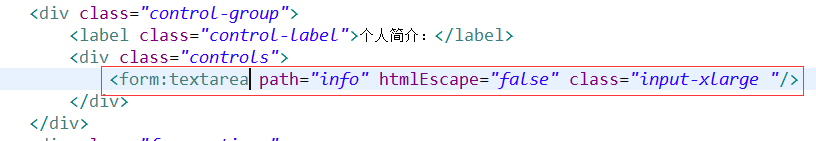
例如这里需要用到富文本



打开testStudentForm.jsp，把



改为

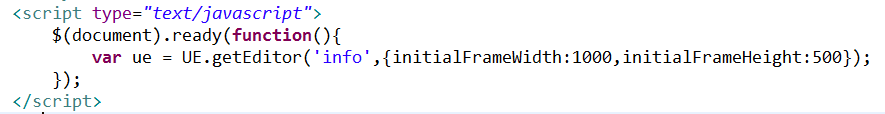


<head>中导入

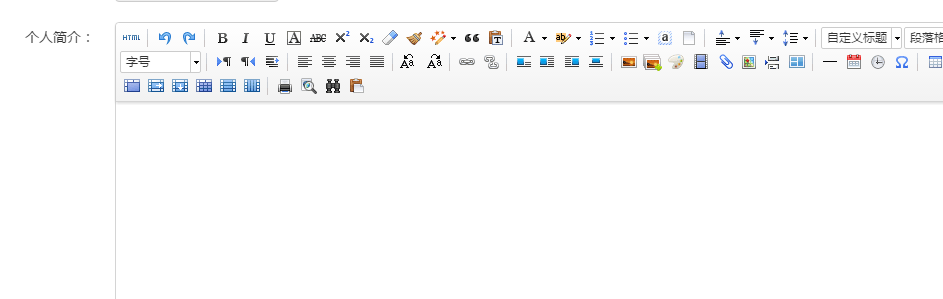
<script type=*"text/javascript"* charset=*"utf-8"* src=*"*${ctxStatic }*/ueditor/ueditor.config.js"*></script>

<script type=*"text/javascript"* charset=*"utf-8"* src=*"*${ctxStatic }*/ueditor/ueditor.all.min.js"*> </script>

加一段script：

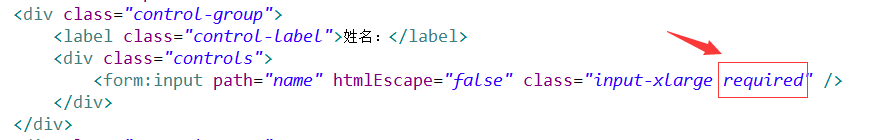


就会变成这样

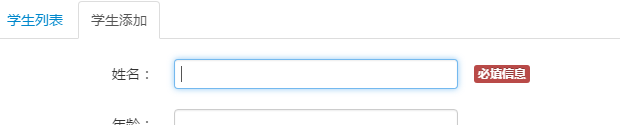


## 7.JQuery-Validation前端校验

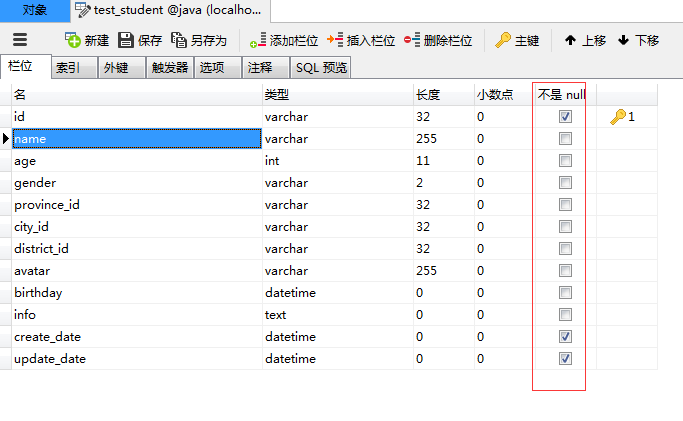
如果页面中某些字段不能为空，在input中添加required类名



这时候如果不填就点击保存就会自动拦截



另外，设计表的时候如果字段是not null



那么生成的jsp中会自动在类名中加上required

## 8.日期控件的使用

文件目录在/static/My97DatePicker

1.导入WdatePicker.js文件，如<script type="text/javascript" src="WdatePicker.js"></script>

2.在<input 中加入如下代码<input class="Wdate" type="text" onClick="WdatePicker({dateFmt:'yyyy-MM-dd HH:mm:ss',isShowClear:false})">即可

WdatePicker.js文件已在style.jsp中导入，使用时<head></head>里面加<%@ include file="/WEB-INF/views/include/style.jsp"%>即可

项目中使用可以复制

<input name="birthday" type="text" readonly="readonly" maxlength="20" class="input-medium Wdate"

value="<fmt:formatDate value="${testPerson.birthday}" pattern="yyyy-MM-dd HH:mm:ss"/>"

onclick="WdatePicker({dateFmt:'yyyy-MM-dd HH:mm:ss',isShowClear:false});"/>

## 9.前端js对密码进行sha1加密：

项目管理后台加密使用了sha1加密，登陆页面一级用户管理中新增用户填请求controller时采取了先加密再请求。加密的js文件在/static/sha/下。使用方法：

1.导入sha.js文件

2.执行加密代码如下

var password = "123456";

var shaObj = new jsSHA("SHA-1", "TEXT");

shaObj.update(password);

password = shaObj.getHash("HEX");

jsSHA这个对象根据不用参数可以实例化不同的sha系列加密对象，如sha-2、sha-256等等，具体请浏览 https://github.com/Caligatio/jsSHA

sha.js文件已在style.jsp中导入，使用时<head></head>里面加<%@ include file="/WEB-INF/views/include/style.jsp"%>即可

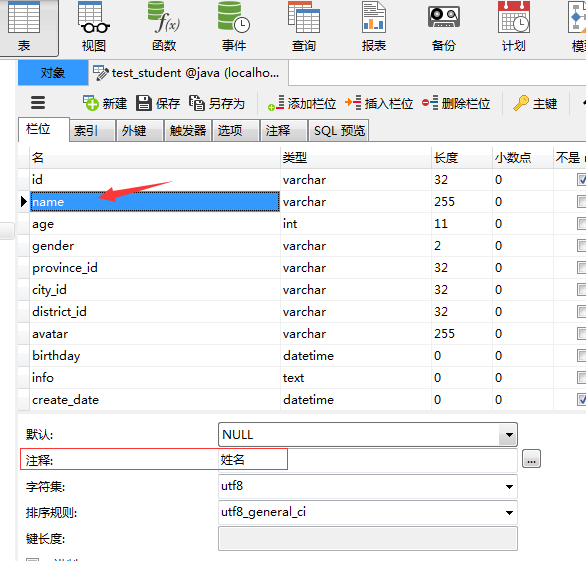
## 10.建表规范

1.每个表必须遵循框架规范包含主键字段并且取名为id，包含创建以及更新时间create\_date、update\_date。

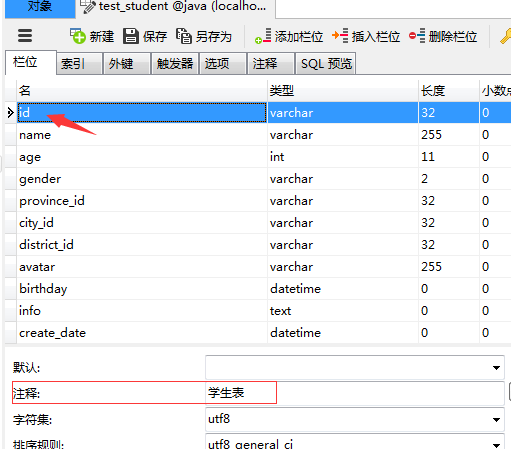
new出来的对象，不需要设置id值，调用save()方法保存的时候如果没有id值，框架会自动补上32位不重复的id，如果是更新操作，会自动更新update\_date字段值，如果是插入操作，会自动补上create\_date和update\_date。

2. eclipse工程名应与表前缀名一致并且不为sys\_

3.注释，每个表的每个字段都应该设置一个注释简单说明字段的意思，如：



注意：其中id的注释应该填写表的名字，如：

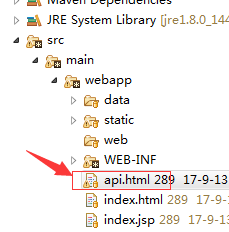


或者为“学生表的id”，不能简单设置为“id”

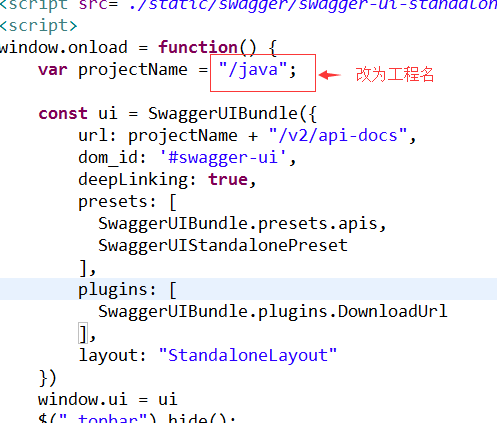
# 第二部分：接口开发

## 1.swagger-ui

文档位置：



1.每个新项目把纯净框架导入eclipse后，需要先修改api.html



把projectName变量设置为工程名

2.接口控制创建应该遵循框架规范创建在com.java.controller.api包中，接口控制器应添加swagger注解让其自动生成文档

3.打开localhost:8080/工程名/api.html



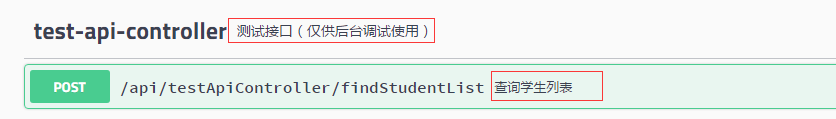
就会出来框架一些本来就写好的控制器

4.访问地址也可以输入ip:端口/项目名/swagger-ui.html，这时候是swagger包中的servlet生成的接口文档地址

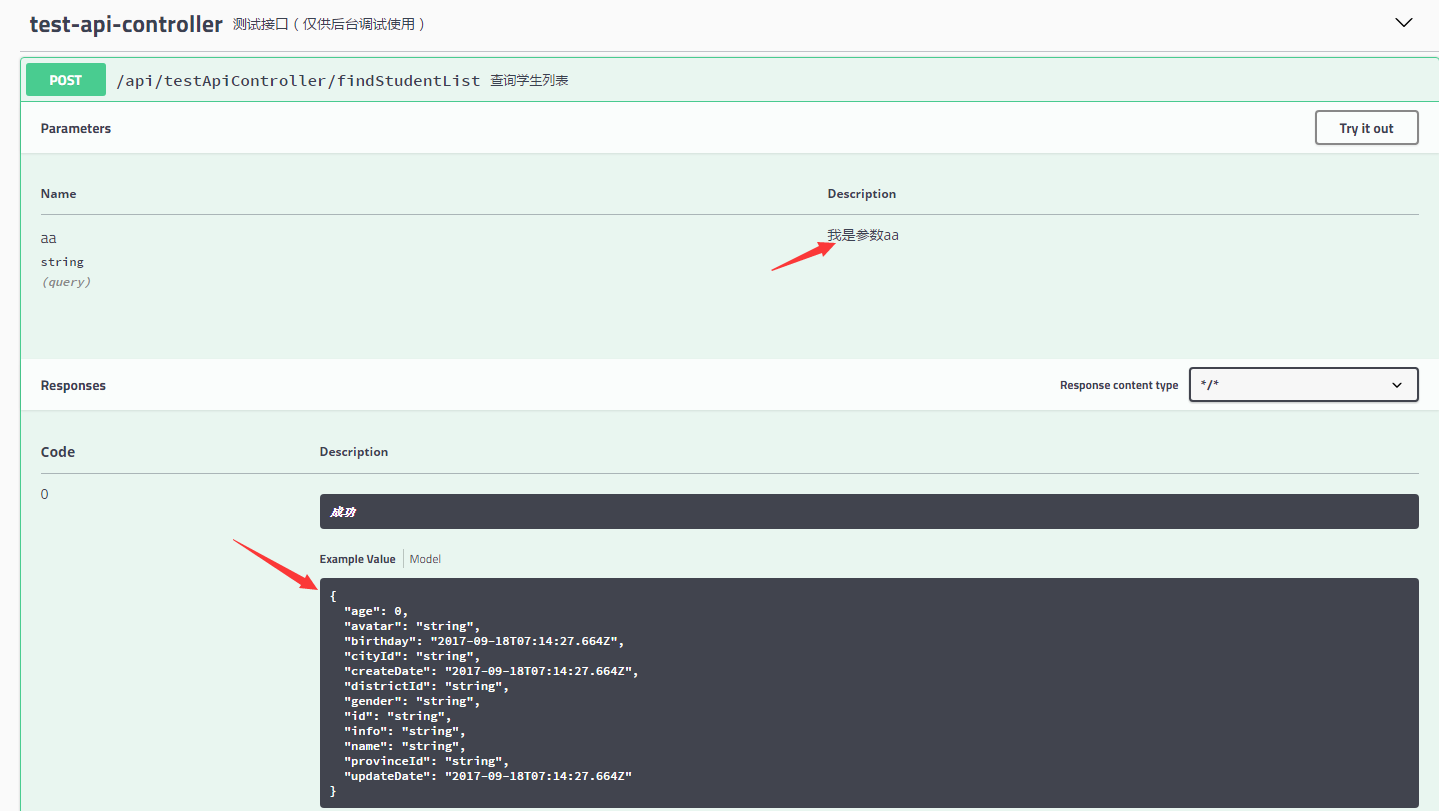
5.创建控制器、写接口



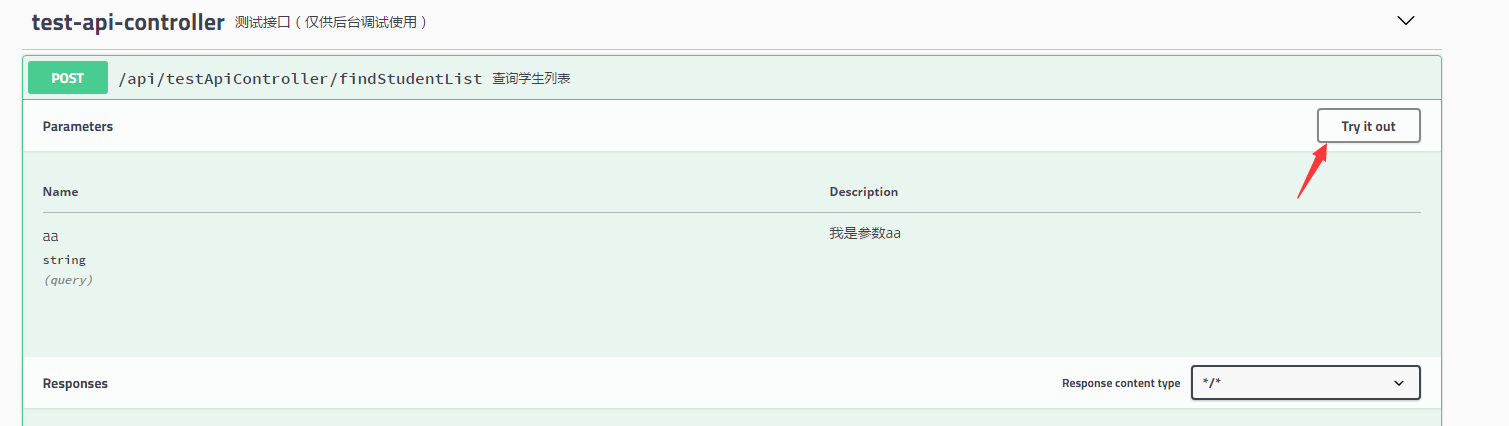
我们来看看文档

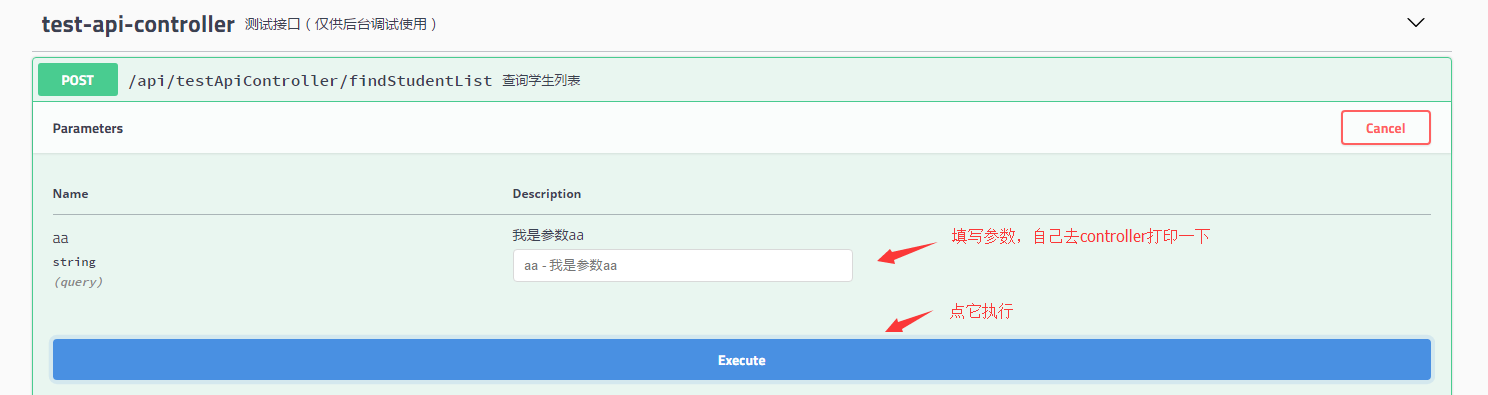


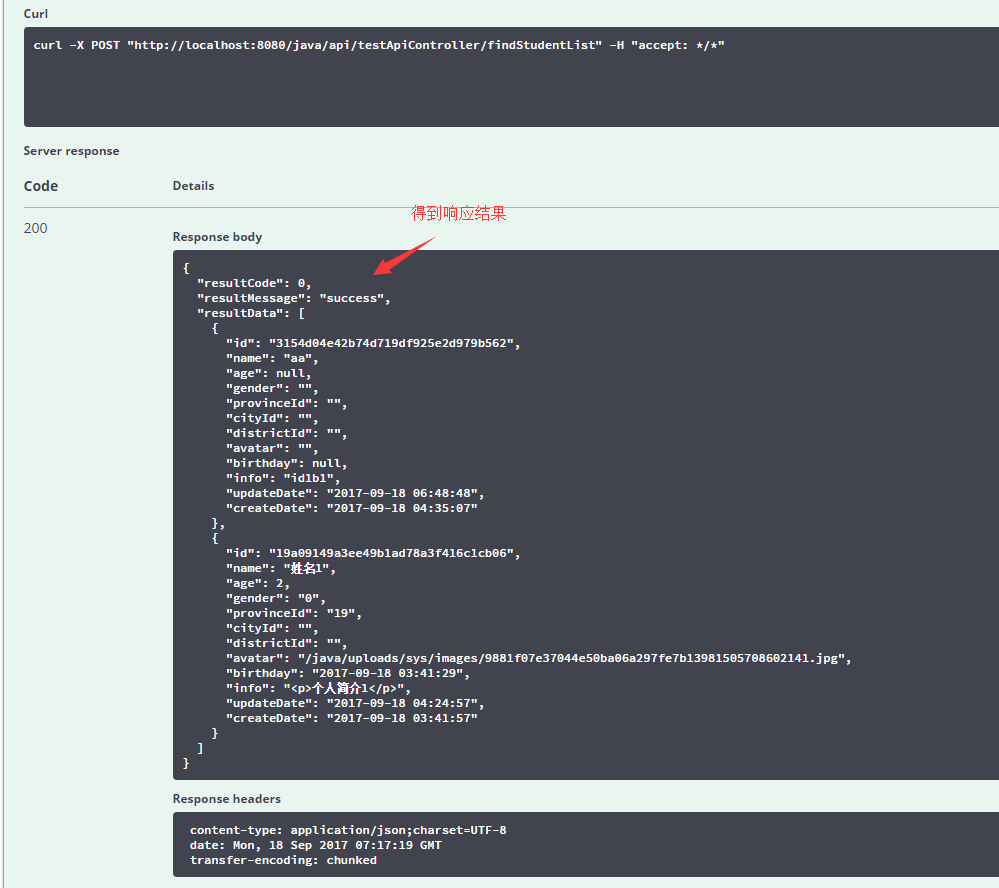
把它展开



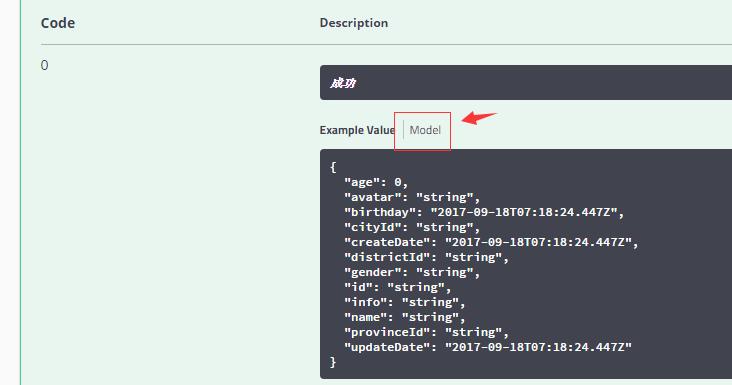
点击try it out可以在线测试

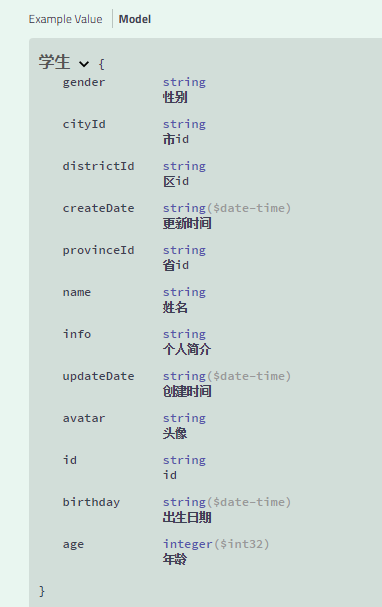




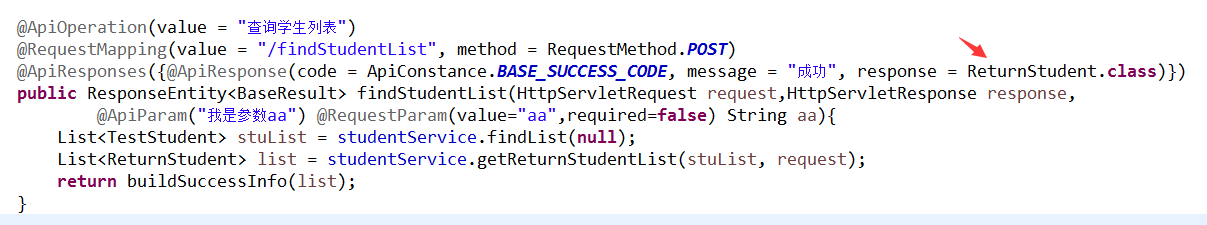


其中，需要强调的是，这里



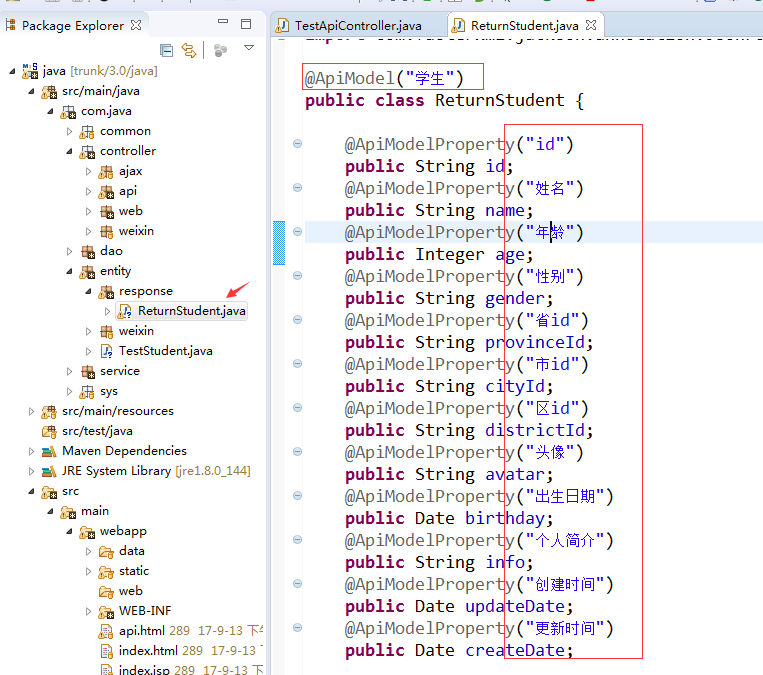


对应的是



为什么返回的是ReturnStudent而不是TestStuden呢？

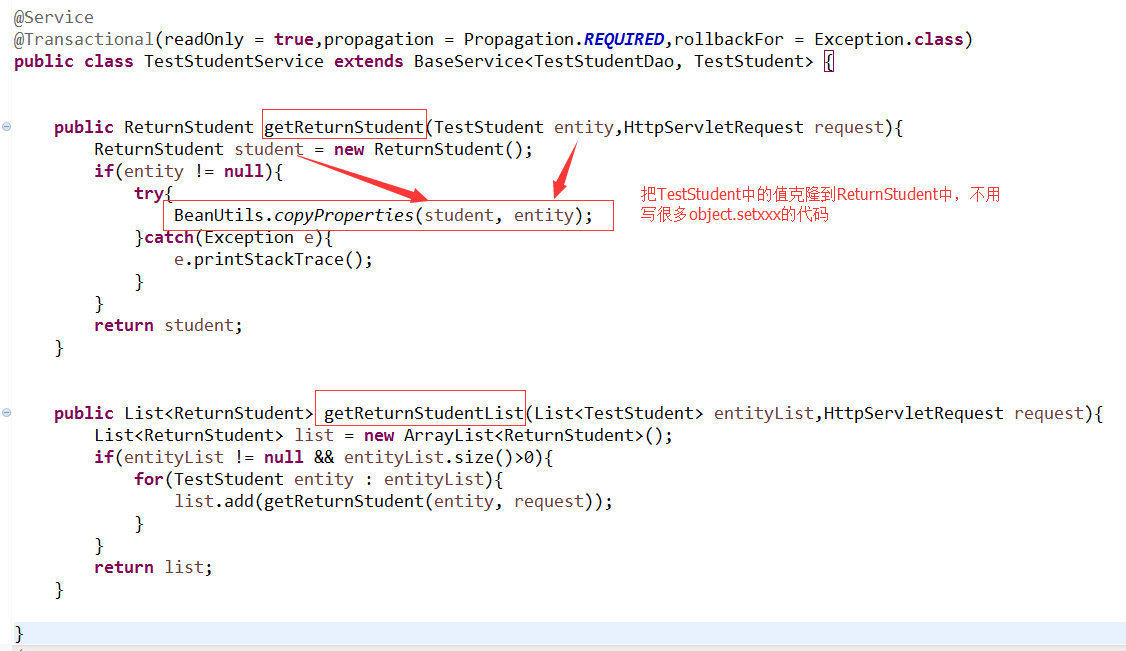
因为每个表对应的实体类都集成了一个父类，父类中有些不必要返回的属性，为了不影响前端筛选，在生成代码的时候都会在com.java.entity.response包中生成一个表对应的返回bean



更多swagger相关访问http://swagger.io，目前没有swagger中文网，了解只能看英文

## 2.响应规范

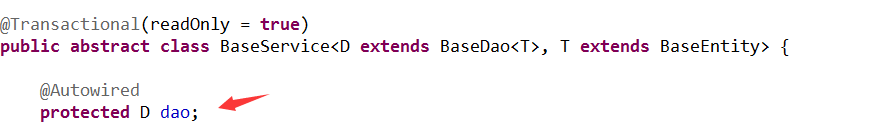
每个接口返回数据表的数据时，应该遵循框架规范，调用对应的getReturn表名()方法来返回数据，在代码生成的时候，service层都会自动生成好两个返回方法，一个用于返回数据详情，一个用户返回数据列表。



## 2.增删查改规范

### 2.1 dao注入：

在生成的service层中，不需要写@Resource xxxDao等等注入代码，框架使用Spring4的泛型注入，看父类：



假设在mybatis的xml中添加了一个方法叫<select id="aa" xxx></select>，dao中写了对应的aa()方法，那么service中直接打dao.aa()即可，所有service已注入对应的dao，并且名字都叫dao，所以打dao.方法名()即可调用dao层的方法

### 2.2 插入和更新

所有插入和更新操作都应该调用service的save()方法

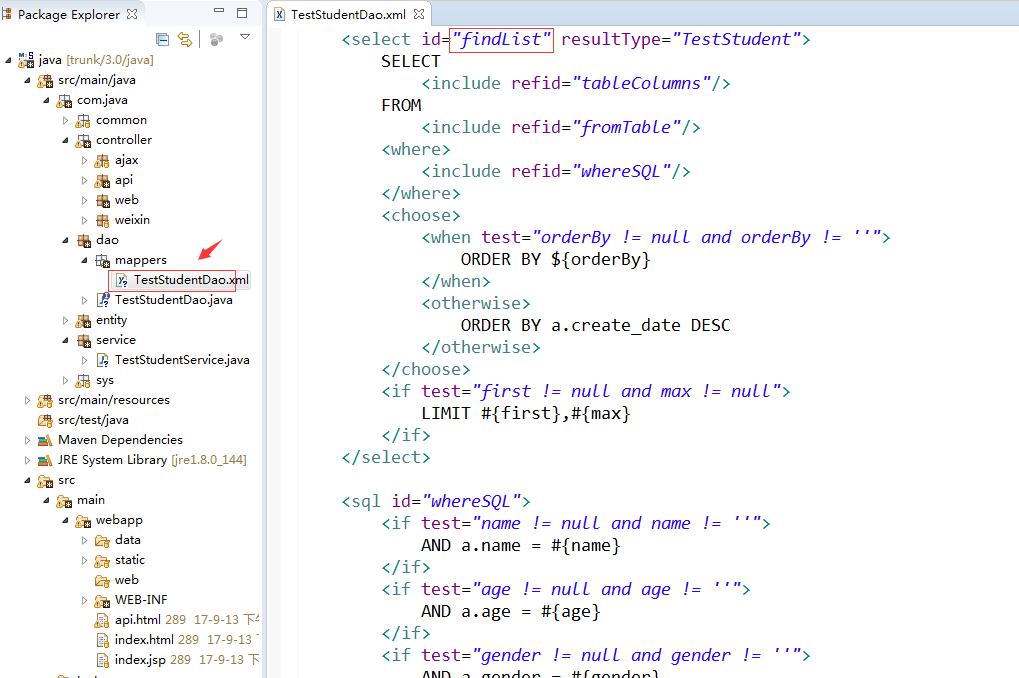
### 2.3 删除

所有删除操作都应调用service的delete()方法

### 2.4 查询

#### 2.4.1 findList

该方法参数是一个实体类，调用时根据参数里面有值得属性查询列表，自己看对应的xml



#### 2.4.2 findListSQL

自己拼写SQL语句

#### 2.4.3 get

参数类型是String，根据id查出单个对象

#### 2.4.4 getBy

根据唯一字段值查出单个对象

### 2.5 分页

在生成的实体类中，都集成了BaseEntity，而各个service的findList参数就是实体类，其中setFirst()和setMax()即可实现分页查询

## 3.文件上传

已经在类Tool（com.java.sys.common.utils）中封装好文件上传方法



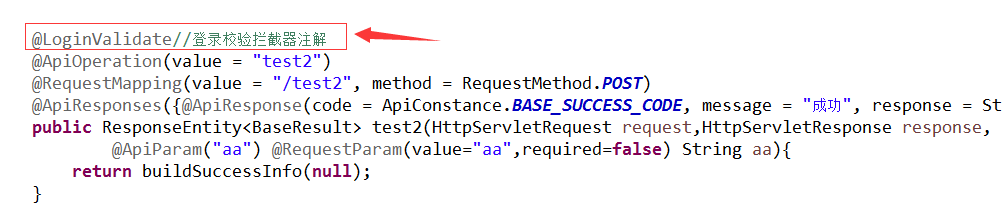
## 4.短信验证码校验规范

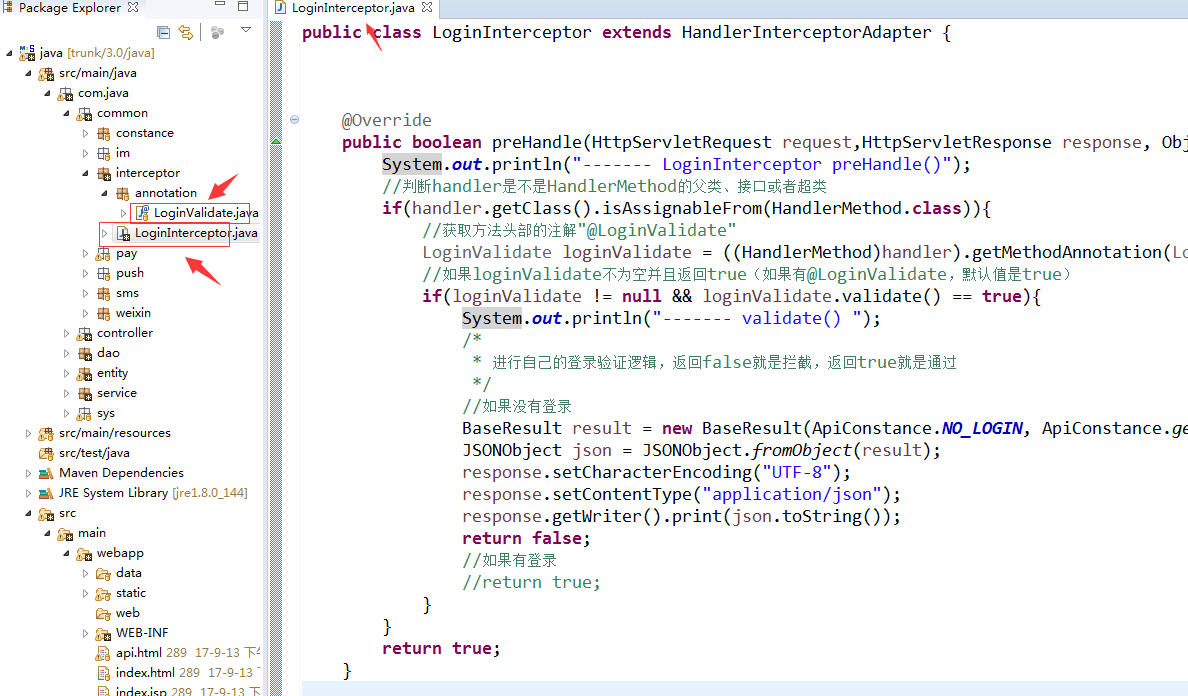
如注册验证码等业务，应该使用EHCache或者Redis把手机号码作为key，把验证码作为value，设定不超过5分钟的过期时间存放在缓存中，在接口中获取参数验证码，对比缓存验证码是否存在、是否一致，从而判断是否合法

# 第三部分：其他功能

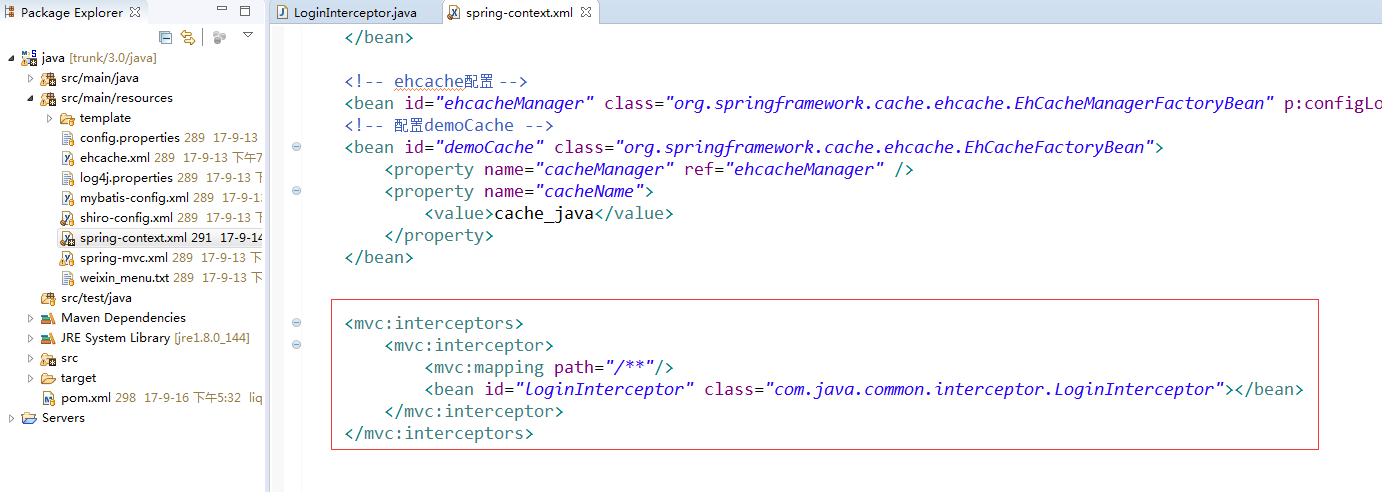
### 1.前置校验（如：登录验证）

项目中往往大量的控制器需要验证是否登录，不应该在每个controller接口中写重复的验证登录代码，可以通过在接口方法头部添加注解从而简化代码，提高阅读性，如：

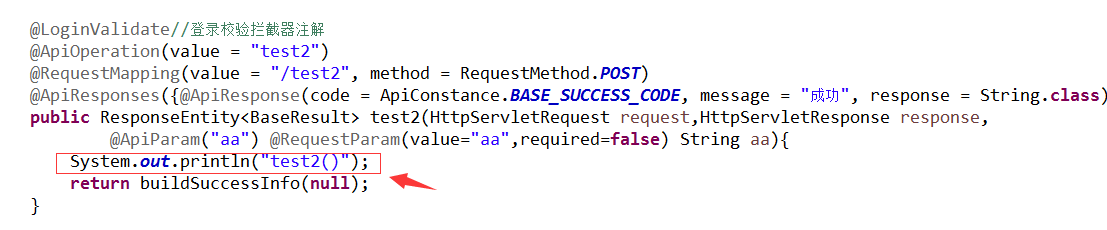


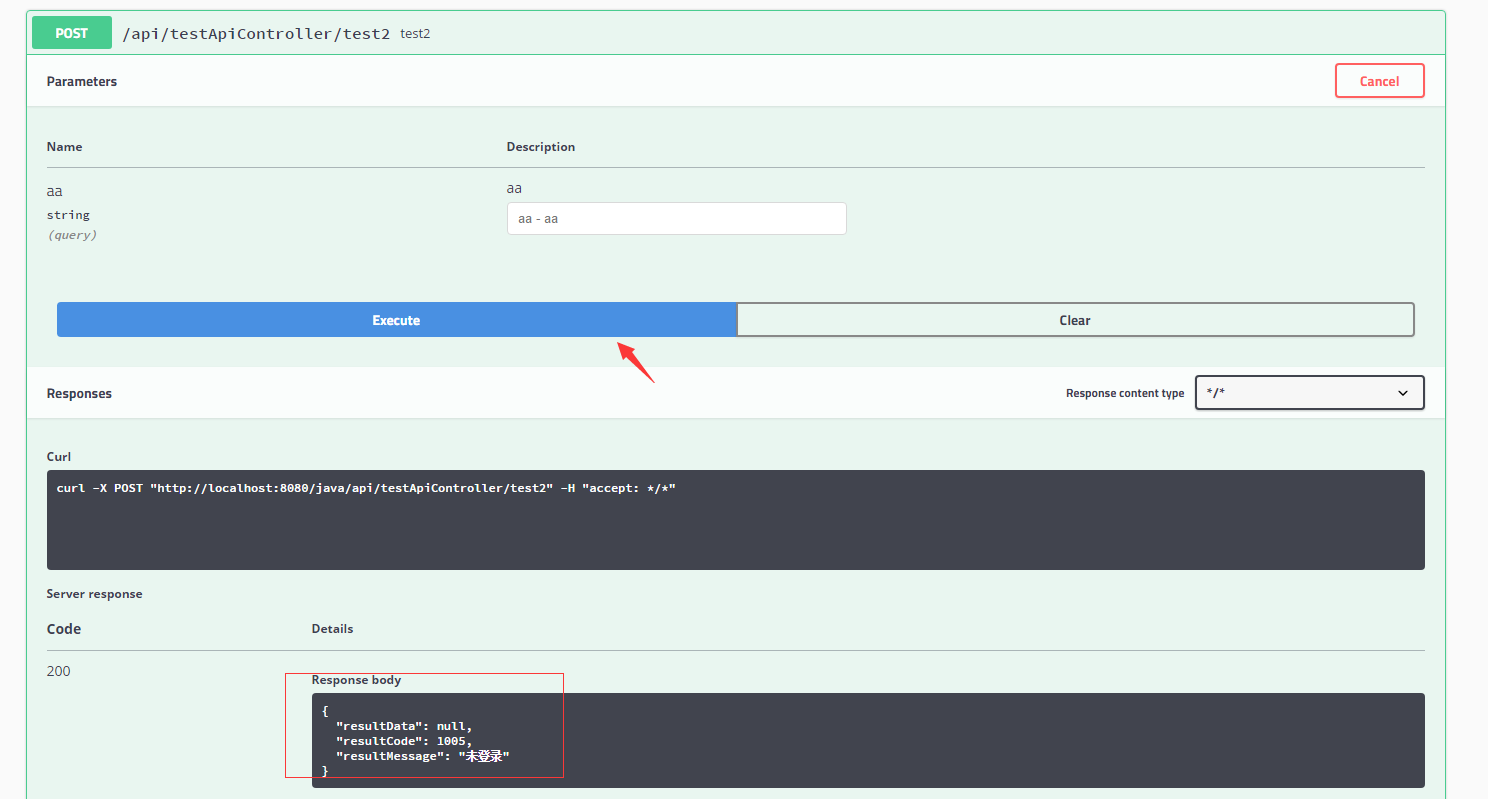


开启拦截器



测试：



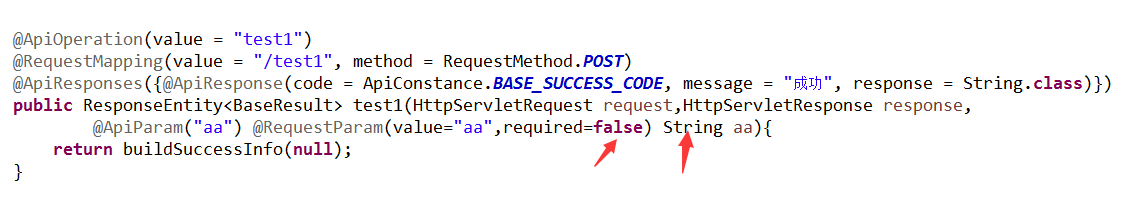


可以看到控制台打印

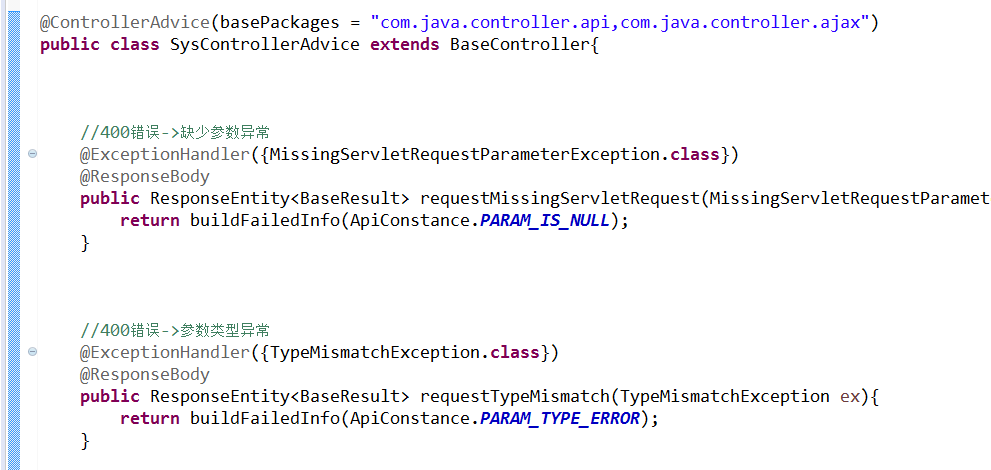


并没有成功进入方法，而是被拦截器拦截并返回对应信息了

### 2.参数校验



如果为required为true，然而请求接口的时候没有传该参数，或者类型是Integer，但是调用者传了“abc”，那么就会进入这个处理类



响应错误信息给调用者，更多检验自己去扩展，在com.java.sys.common.advice包中

### 3.全局异常处理

在com.java.sys.exception包中有一个全局异常处理类，已经写了基本的异常处理，更多处理自己去修改并且扩展

### 4.支付宝、微信支付封装

支付宝、微信支付的demo已经集成在com.java.controller.ajax的PayAjaxController和com.java.controller.api的PayApiController中，自己去看。其中PayAjaxController主要包含支付宝PC网站支付、支付宝H5网站支付、微信公众号支付、微信扫码支付等；而PayApiController中包含支付宝APP支付、微信APP支付

### 5.微信公众号开发封装

微信公众号开发相应demo已经集成在com.java.controller.weixin包的controller里面，自己去看

### 6.SQL监控（druid）

框架集成的druid连接池，可以通过项目路径/druid/访问SQL监控页面，此路径做了权限拦截，超级管理员才能登陆，否则会跳到后台登陆页面，登录成功后再访问该路径即可

### 7.EHCache缓存

框架已集成并封装好EHCache缓存，在com.java.sys.common.cache包中，自己去看并使用