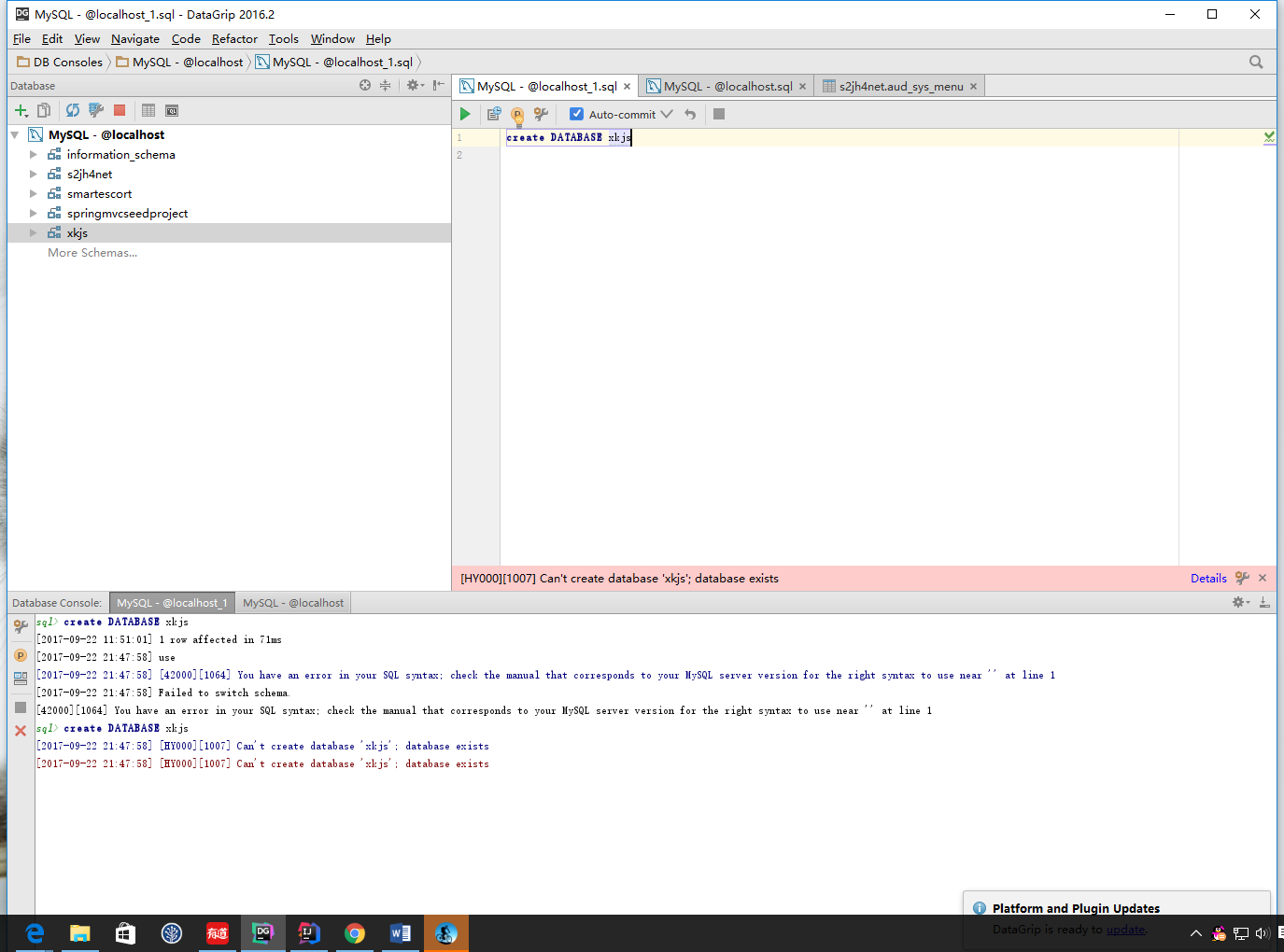
2017/9/22

删除数据库mysql语句 delete database xkjs；

创建新的数据库 create database xkjs;



Cache缓存.

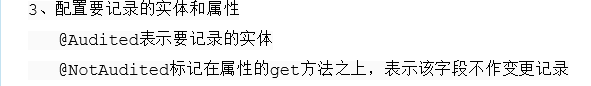
2017/9/23

1.install Idea Lombok plugins插件**在 java平台上，lombok 提供了简单的注解的形式来帮助我们消除一些必须有但看起来很臃肿的代码, 比如属性的get/set，及对象的toString等方法，特别是相对于 POJO;**

出现的问题：没有error但有波澜线。

2. @Setter：注解在属性上；为属性提供 setting 方法  
      @Getter：注解在属性上；为属性提供 getting 方法、

3. Hibernate缓存配置   
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_   
Hibernate的缓存分为：   
　　一级缓存：在Session级别的，在Session关闭的时候，一级缓存就失效了。   
　　二级缓存：在SessionFactory级别的，它可以使用不同的缓存实现，如EhCache、JBossCache、OsCache等。   
　　   
缓存的注释写法如下，加在Entity的java类上：   
　　@Cache(usage = CacheConcurrencyStrategy.NONSTRICT\_READ\_WRITE)   
　　   
缓存的方式有四种，分别为：   
　　CacheConcurrencyStrategy.NONE   
　　CacheConcurrencyStrategy.READ\_ONLY，只读模式，在此模式下，如果对数据进行更新操作，会有异常；   
　　CacheConcurrencyStrategy.READ\_WRITE，读写模式在更新缓存的时候会把缓存里面的数据换成一个锁，其它事务如果去取相应的缓存数据，发现被锁了，直接就去数据库查询；   
　　CacheConcurrencyStrategy.NONSTRICT\_READ\_WRITE，不严格的读写模式则不会的缓存数据加锁；   
　　CacheConcurrencyStrategy.TRANSACTIONAL，事务模式指缓存支持事务，当事务回滚时，缓存也能回滚，只支持JTA环境。

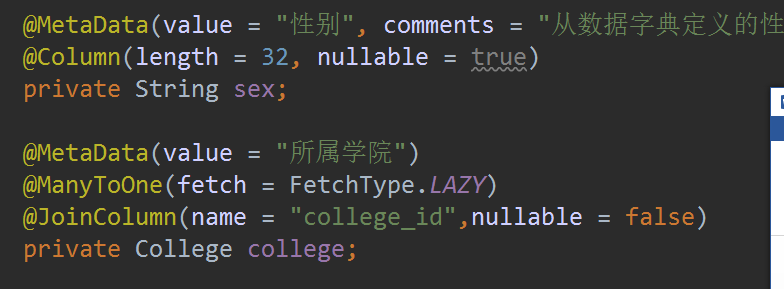
4. @audited注解

5. JPA定义实体之间的关系有如下几种：

举例entity里边student，一个学生对应一个用户故事one to one，很多学生对应一个专业，一个班级，一个学院so，是many yo one。  
@OneToOne   
@ManyToOne   
@OneToMany   
@ManyToMany   
在定义它们的时候可以通过fetch属性指定加载方式，有两个值:   
FetchType.LAZY：延迟加载   
FetchType.EAGER：急加载

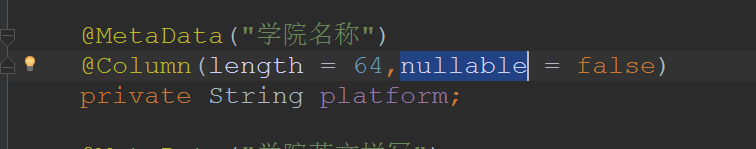
[6.@Column](mailto:6.@Column)而@Column标注的是表中不包含表关系的字段。比如说性别就是这个学生表里边单独的字段，而底下的所属学院还有一张单独的表，表名主键叫“college\_id”

@JoinColumn注释的是保存表与表之间关系的字段，它要标注在实体属性上。

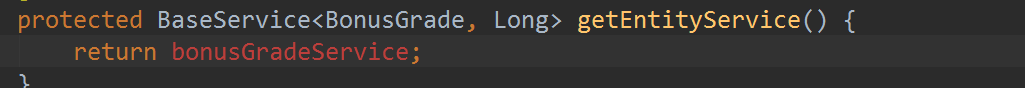


6. unique=true是指这个字段的值在这张表里不能重复，所有记录值都要唯一，就像主键那样

entity里边设置如下图一！！！！！！这个@Colnumn(length=64,nullable=false)是这个字段在保存时**必需有值\*（图二）**，不能还是null值就调用save去保存入库  
用法是不同的

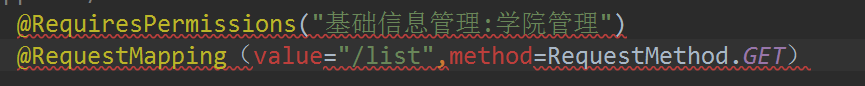
，

7.return后面第一个字母要小写



8.ctrl+alt+L规范代码快捷键

9.谁能看出来是括号的问题，下面那个括号有问题！



（2017/9/24）

1. @ResponseBody 将内容或对象作为 HTTP 响应正文返回，

使用@ResponseBody将会跳过视图处理部分，而是调用适合HttpMessageConverter，将返回值写入输出流。

[2.@OperationResult](mailto:2.@OperationResult)

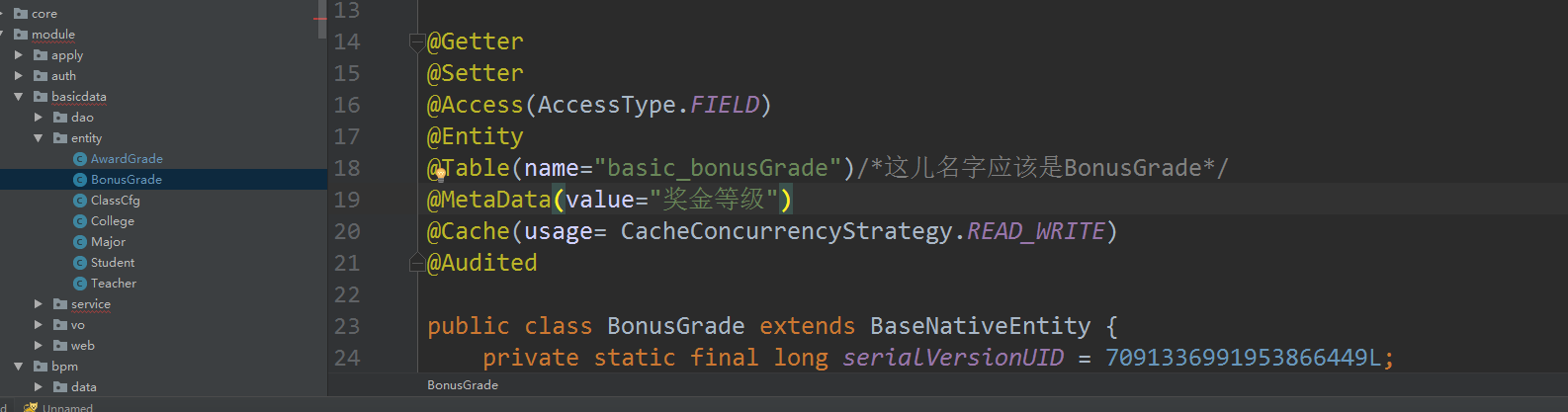
@RequestParam()

3. Can't start Git:git.exe问题的解决

4.注意！！！改的话不仅是路径，还有对应的字段也要改！

Entity、

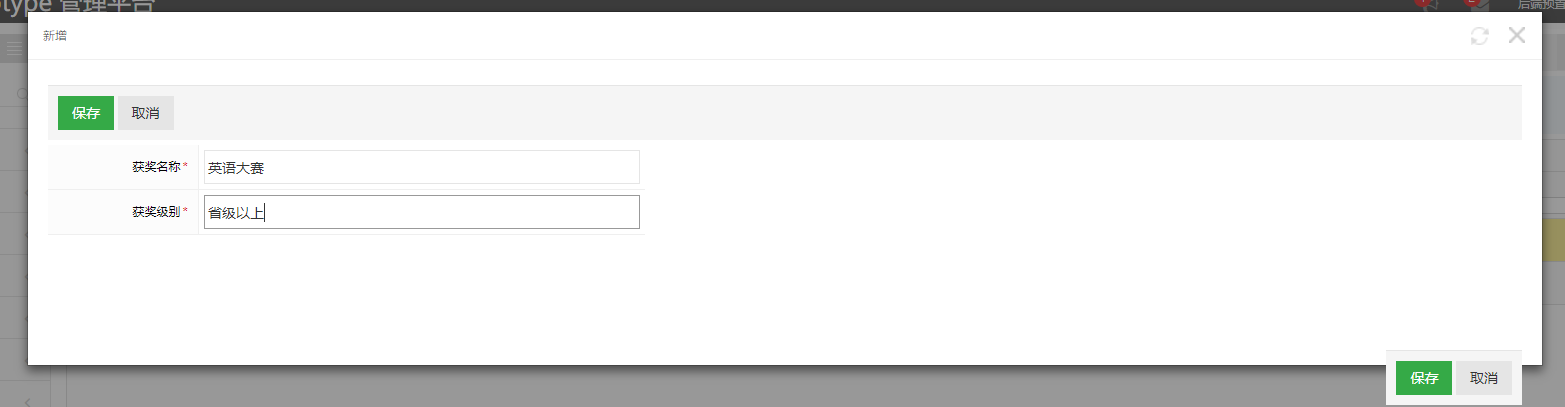
5.这儿的名字是数据库里边的表名字。注意咯



6.找不到错误。。重新写一遍

7.Ctrl+Tap 来回切换

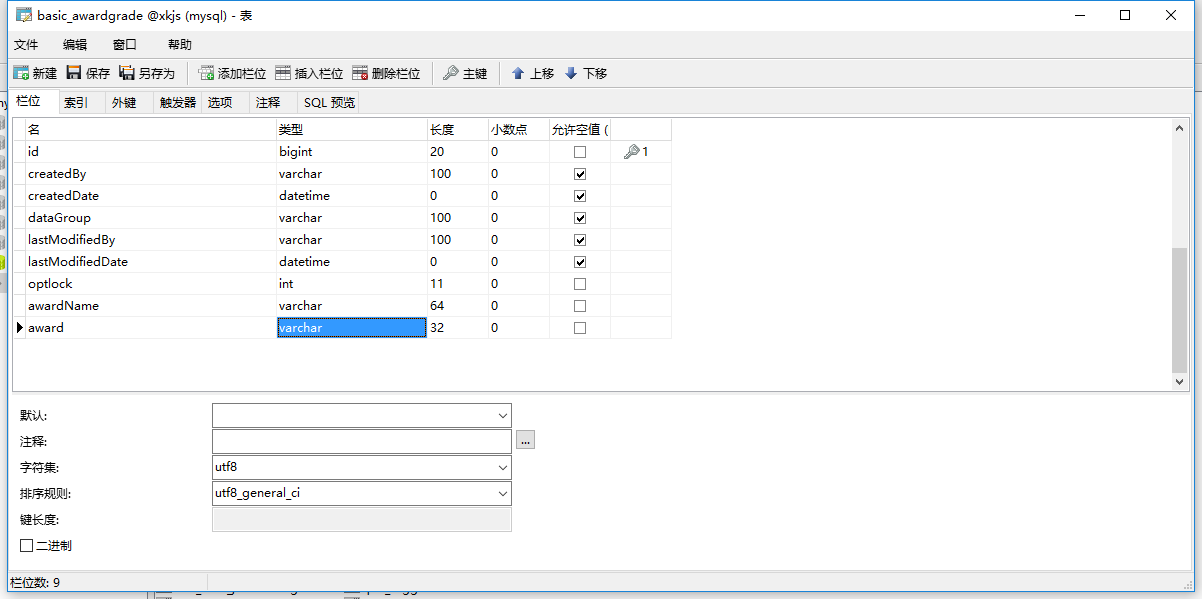
8mysql点保存的时候数据异常了，因为这是个必填字段，在之前的时候把entity里边



显示数据异常！

改了一下名字，之前是字段名是。。AwardGrade与entity里边重名了所以吧字段改成了awardLever。注意！！是必填所以再改之前mysql数据库的表里边就有这个\*字段了，故不能保存，因为需要填的是三个必填字段！你只填了两个！

解决方法：打开mysql那张表→设计表→删除之前AwardGrade在这一行即删除栏位。再次运行 酒酒ok了，问题解决。



9．多个省级级别对应一个获奖名称，ManyToOne应该是

(2017/9/27)

1.调取数据库的表删掉（删了两个表之后就可以调取一张表了）

2.[PO](https://www.baidu.com/s?wd=PO&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1Y3P1DzPjTYPyDsmyD3n1Ds0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EPHm3nW0zPjfL):persistant object 持久对象,可以看成是与数据库中的表相映射的java对象。使用Hibernate来生成[PO](https://www.baidu.com/s?wd=PO&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1Y3P1DzPjTYPyDsmyD3n1Ds0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EPHm3nW0zPjfL)是不错的选择。   
3.VO:value object值对象。通常用于业务层之间的数据传递，和[PO](https://www.baidu.com/s?wd=PO&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1Y3P1DzPjTYPyDsmyD3n1Ds0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EPHm3nW0zPjfL)一样也是仅仅包含数据而已。但应是抽象出的业务对象,可以和表对应,也可以不,这根据业务的需要.

entity里的每一个字段，与数据库相对应，而dto里的每一个字段，是和你前台页面相对应，而VO，这是用来转换从entity到dto，或者从dto到entity的中间的东西。

解释：你的html页面上有三个字段，name，pass，age  
你的数据库表里，有两个字段，name，pass(注意没有age哦)  
  
而你的dto里，就应该有下面三个(因为对应html页面上三个字段嘛)  
private string name；  
private string pass;   
private string age;  
这个时候，你的entity里，就应该有两个(因为对应数据库表中的2个字段嘛)  
private string name；  
private string pass;  
到了这里，好了，业务经理让你做这样一个业务“年龄大于20的才能存入数据库，（pass）否则不让”  
这个时候，你就要用到vo了，你要先从页面上拿到dto，然后判断dto中的age是不是大于20，如果大于20，就把dto中的name和pass拿出来，放到vo中，然后在把vo中的name和pass原封不懂的给entity，然后根据entity的值，在传入数据库，这就是他们三个的区别  
  
PS，VO和entity里面的字段应该是一样的，vo只是entity到dto，或者dto到entity的中间过程，如果没有这个过程，你仍然可以做到增删改查，这个就根据具体公司来的，纯手打，望采纳。

<http://blog.csdn.net/dac55300424/article/details/17533761>

# (2017/9/28) @Accessors

1. 一个为getter和setter设计的更流畅的API.（可以查查lombok@accessors注解<http://blog.csdn.net/cc_smile0702/article/details/73521878>）

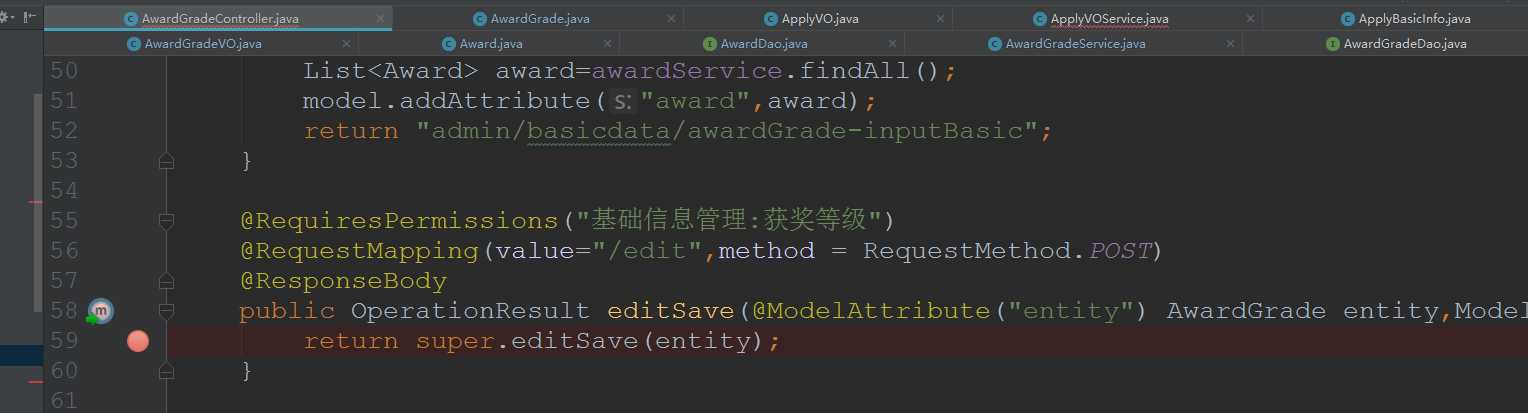
① fluent 一个布尔值。如果为真，pepper的getter就是 pepper()，setter方法就是pepper(T newValue)。并且，除非特别说明，chain默认为真。  
 ② chain 一个布尔值。如果为真，产生的setter返回的this而不是void。默认是假。如果fluent=true，那么chain默认为真。

(2017/9/29)很瞌睡！

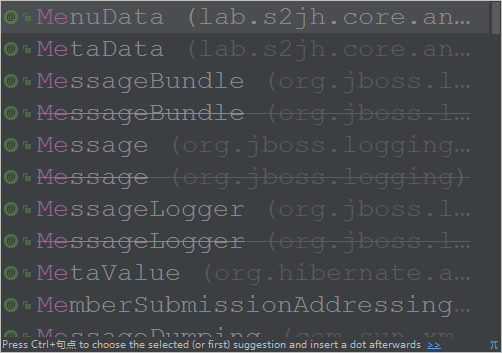
@Autowired 注释，它可以对类成员变量、方法及构造函数进行标注，完成自动装配的工作。 通过 @Autowired的使用来消除 set ，get方法。

(2017/9/30)vo

使用外键→这儿不能保存实体→需要用到vo那么



（2017/10.2）MetaData与Menudata的区别：



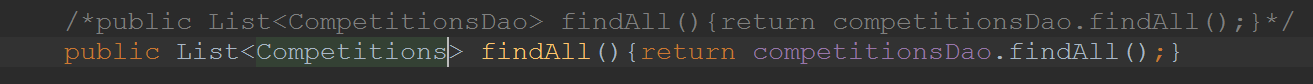
（2017/10/3）

1. List<T>类表示可通过索引访问的对象的强类型列表，提供用于对列表进行搜索、排序和操作的方法。

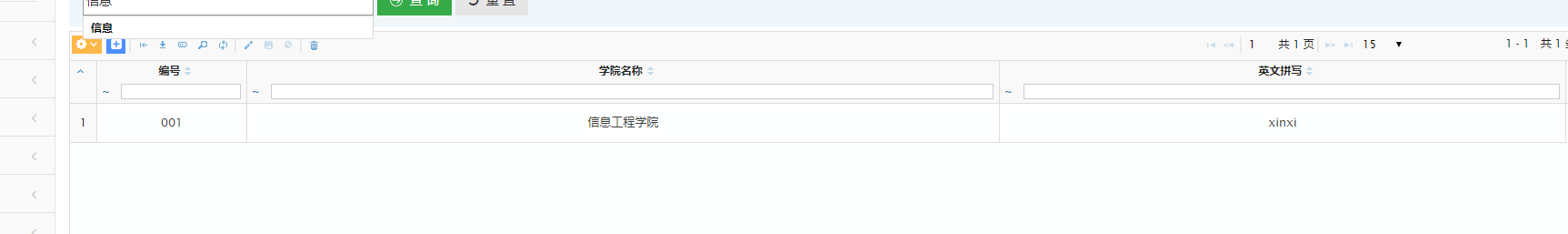
如下图这个是CompetitionsService（业务层）这里边要调用Dao的方法，很显然

public List<列表名> findAll(){return ——Dao.findAll();}

这里的列表名是实体的名字即entity的名字Competitions

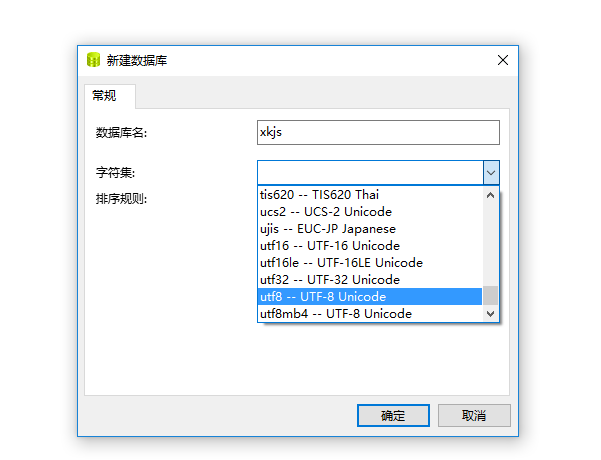
2.Jsp搜索

<input type=”type” name=”search[‘CN\_name\_OR\_platform’]” placeholder=”名字，学院。。”这样就可以查出来了。

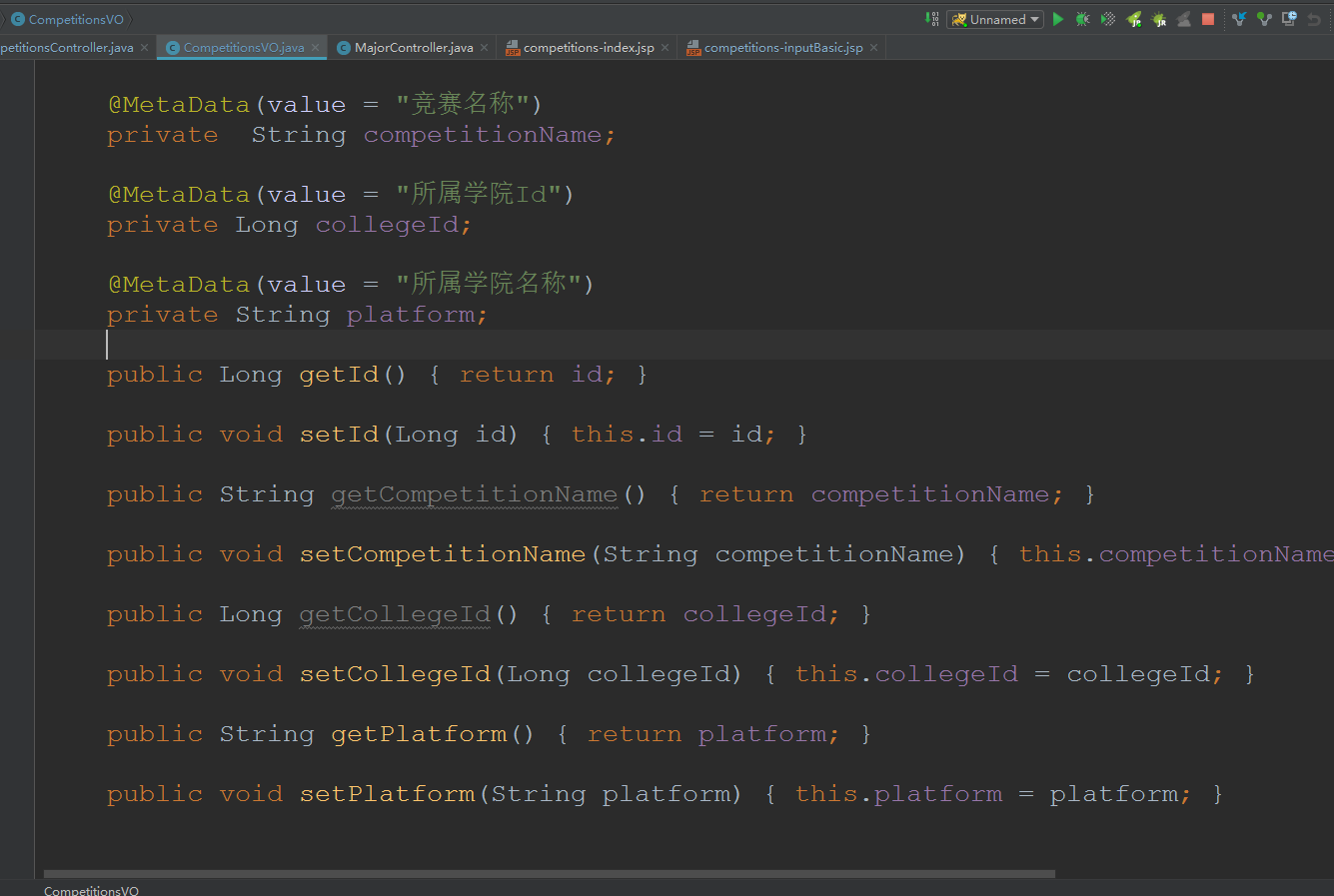


（2017/10/4）

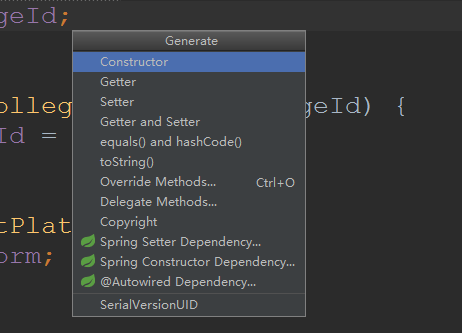
1. 新建数据库 右击写上数据库名，字符集选为Utf-8，排序规则不用动。



1. refactor name 全部可以换掉（lever→level）
2. stage all 点开文件可以看到自己修改的是哪里，描述一下自己修改的部分 先提交，再推送。
3. 500错误内部代码有问题。If{ }
4. 底下这个VO 里边的方法不需要手动写!

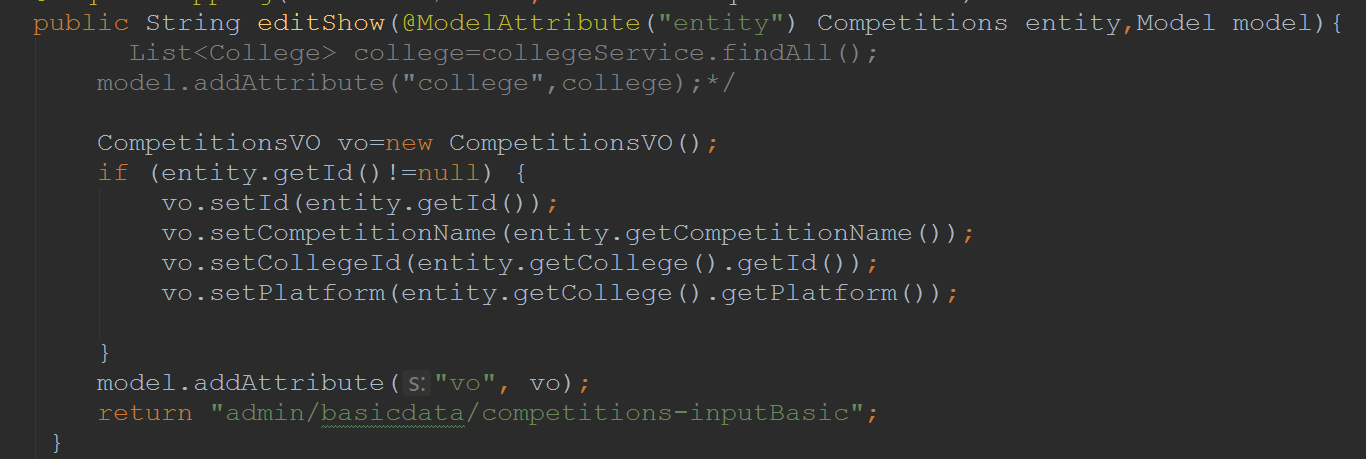


这里用到Alt+insert键既可以自动生成方法。VO里边getter和setter方法。



6用VO调用选择的表外键。（实现编辑的功能。）

1. 将entity的值存在vo里边，定义一个vo，方便处理。



1. Ⅰ.改变Jsp的路径，里边competitions-inputBasic.jsp中，原本是如下：

<form:input type=”hidden” path=”college.id” id=”platform” hidden=”true”/>

<foem:input path=”college.platform” id=”college” readonly=”true”/>

把path改咯，改为：

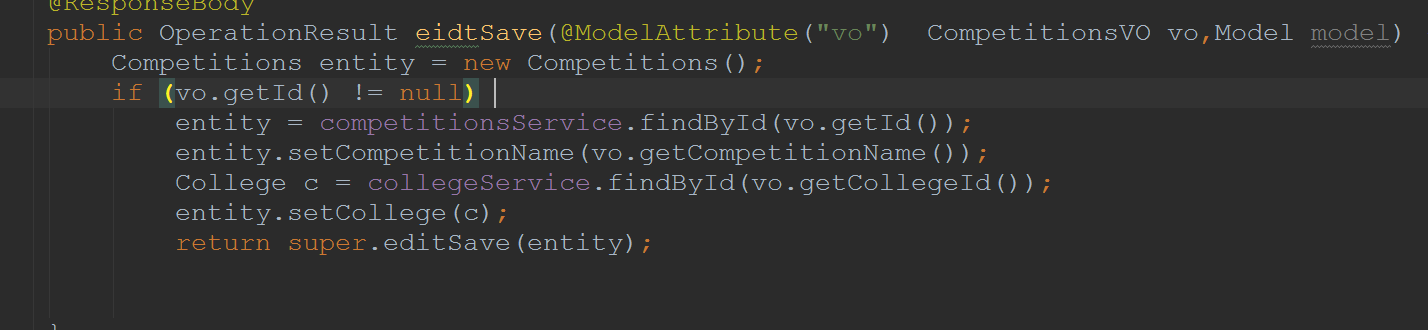
<form:input type=”hidden” path=”collegeId” id=”platform” hidden=”true”/>

<foem:input path=”platform” id=”college” readonly=”true”/>

Ⅱ.将 competitions-inputBasic.jsp中，原本是entity改为vo。



1. 再将vo转回entity保存在数据库里边。



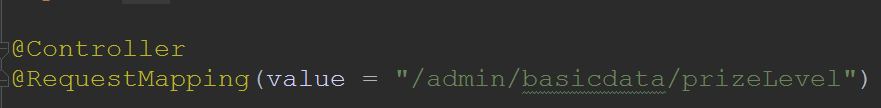
Ok了！

7. readonly="true"

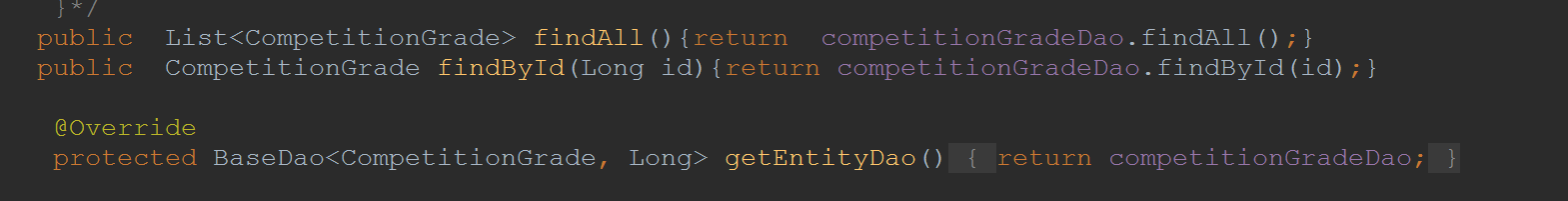
readonly 属性规定文本区为只读。  
在只读的文本区中，无法对内容进行修改(即直接引用学院的信息)，但用户可以通过 tab 键切换到该控件，选取或复制其中的内容。

(2017/10/6)

1.获奖等级页面，一直转圈，还不报错，。。。最后是prizeLevel这写错了！！！！！！！



1. 写Service调用Dao里边的方法return competitonDao.findAll(); 斜杠/后边return后边都得用小写！！！



小插曲：关系**表示层→controller→service→Dao（在service里边new一个Dao类的对象，调用Dao类方法访问数据库）**：

数据访问层

业务层

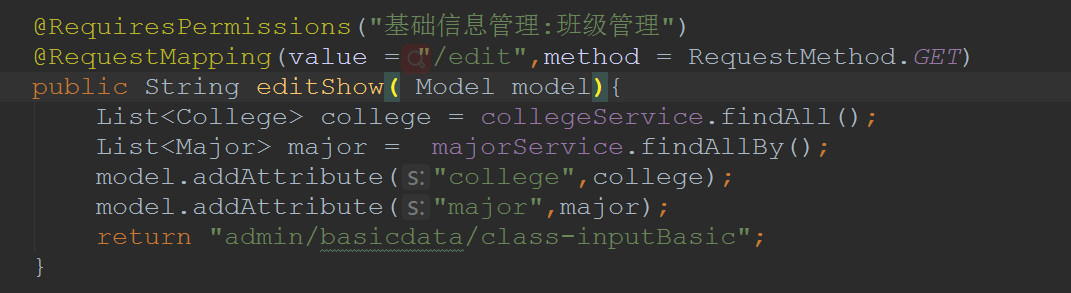
写的时候要：先entity-dao-service（controller也要改）。

1. 页面404出不来页面加载失败。那就是页面与controller有问题！！！！

找了半年原来是jsp的名字competition的名字多写了一个i 写成了competiition哎！

2017/10/7

1.班级controller之前

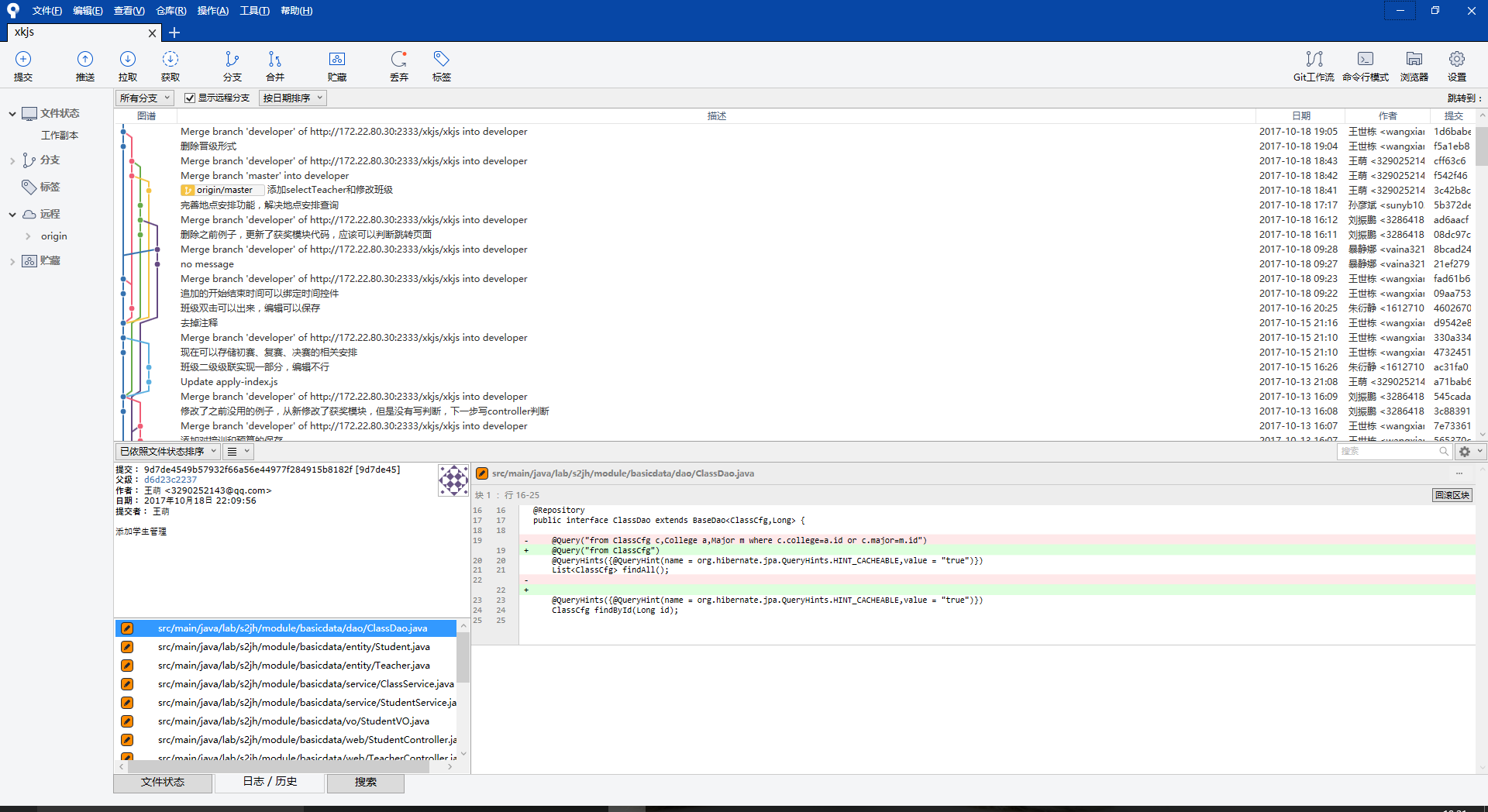


2.唯一标识，一个专业只能出现一次

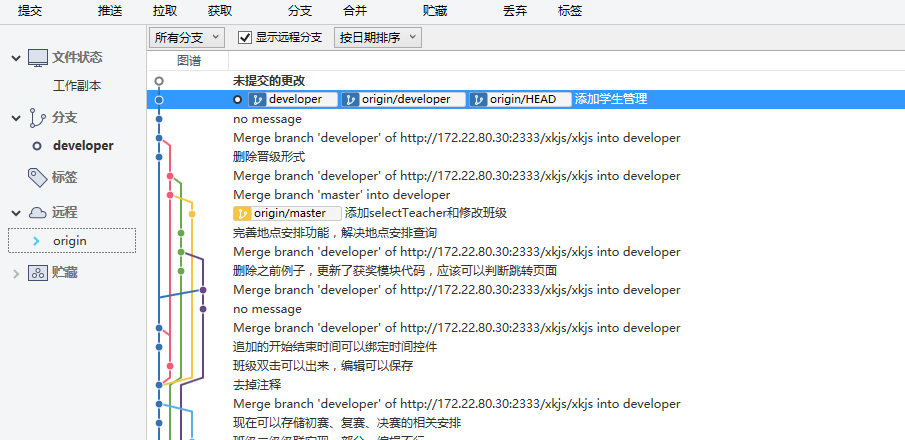
@JsonIdentityInfo(generator = ObjectIdGenerators.PropertyGenerator.class, property = "id",scope = Competitions.class)

1. [http://](http://172.22.80.30:2333/xkjs/xkjs.git)[172.22.80.30:2333/xkjs/xkjs.git](http://172.22.80.30:2333/xkjs/xkjs.git)

拉取之后手动输入和复制不一样，每回都要

1. <http://Frozen@gitlab/xkjs/xkjs.git> zyj12231
2. ①每次重新导入项目需要先删E:/xkjs这个文件夹和数据库(把需要的备份一下) 

②在sourcetree里边新建克隆，执行4，确定→检出（右击建立分支developer）如下图；

③点开idea根据idea整合，导入项目改变文件字符类型（如下图）；



④配置服务器，热部署更改一下为下面这个，重建数据库，启动项目ok了；



（2017/10/12）

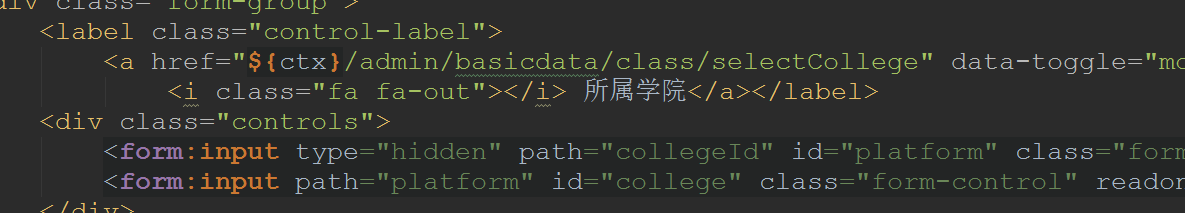
1. sourceTree，拉取代码时有错误提示：

error: Your local changes to the following files would be overwritten by merge:

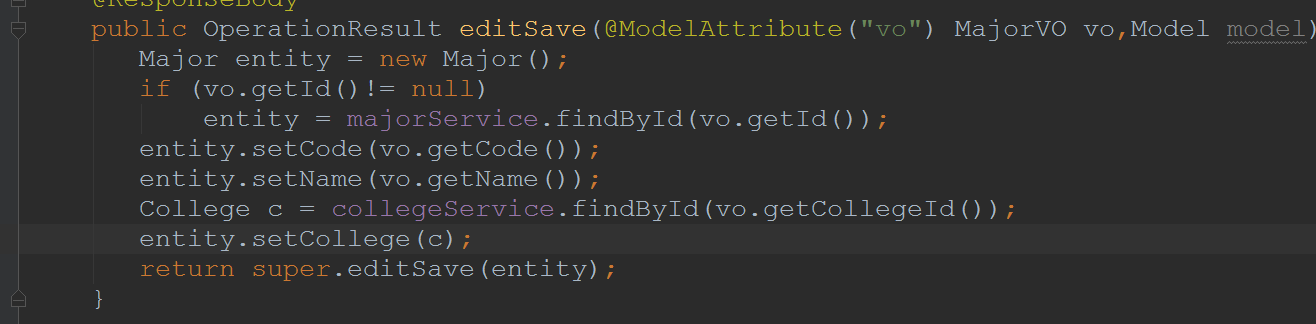
s2jh4net.iml

解决方案：先暂存自己的，拉取别人的丢弃s2jh4net.iml这个文件其余暂存。先不推送(到服务器)可以最终改完再提交推送上去。

1. 班级管理；因为调取学院的时候是一个链接所以MajorController里边vo就很简单了



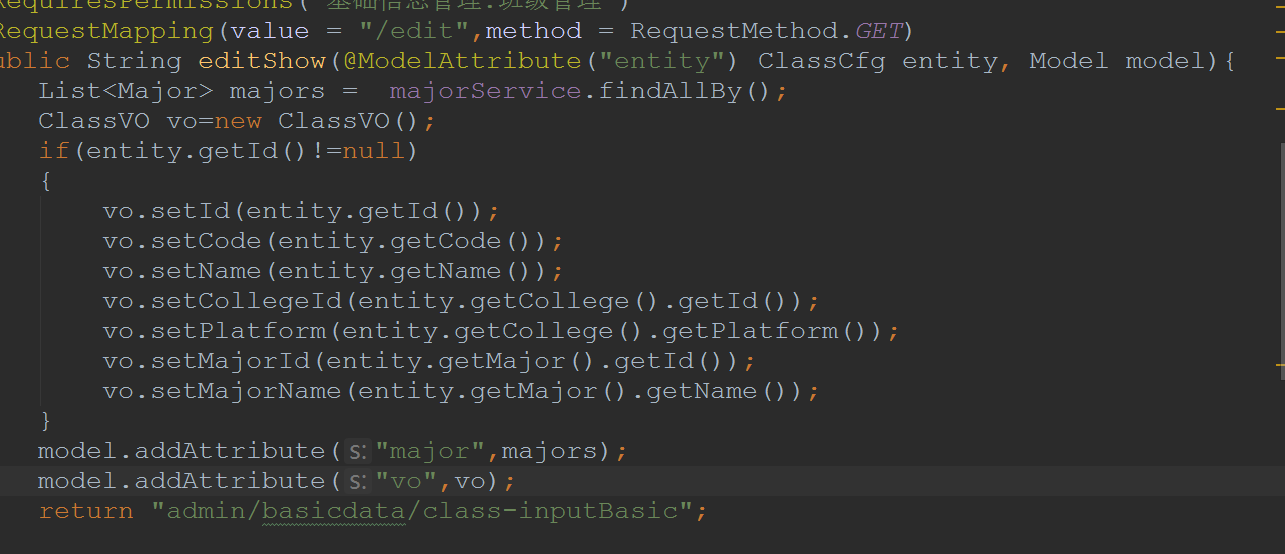
这是class-index.jsp,解决一个点击选取学院出不来，点了专业之后再点才可以出来，

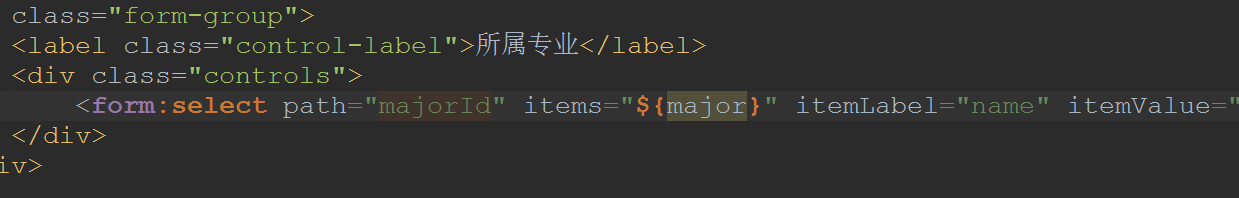


但调其他的时候（比如专业，获奖等级。。）的时候就需要传值得

List<Major> majors =majorService.findAllBy();//这里的majors是随便起的，找到这一列表所有专业。

Model.addAttribute(“major”,majors)//往jsp传值





这里必须是majorId这儿传的是id而不是名字！！！

(2017/10/13)

1. <http://uule.iteye.com/blog/799793>

jquery笔记

1. <http://blog.csdn.net/u011526599/article/details/51340726>
2. http://blog.csdn.net/u011526599/article/details/51340726

二级联动示例。呵呵呵、、、

1. request.getContextPath()应该是得到项目的名字,如果项目为根目录,则得到一个"",即空的字条串, 如果项目为abc, <%=request.getContextPath()% >/ 将得到abc/,[服务器端](https://www.baidu.com/s?wd=%E6%9C%8D%E5%8A%A1%E5%99%A8%E7%AB%AF&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YLPW03n1u-PWT1mHDkm1m40ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EPH0knWTvP1mk)的路径则会自动加上

(2017/10/14)

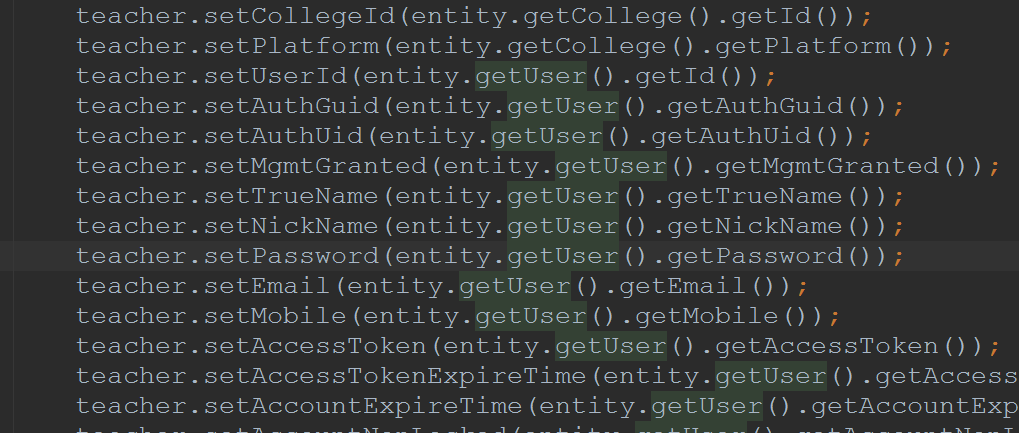
1. 用idea写一个简单的java小程序必须写全这句,否则运行不出来

public static void main(String[] args){ }



(2017/10/15)

1. js，jquery里边打印alert()测试，有没有响应。
2. 这里通过getUser()赋予这个老师这些信息，学院为什么getuser().是不是因为这个老师就本来就是这个学院的，user是一个单独的表



1. 点学生选择班级数据可以点但数据出不来在class-select，这是因为这个方法回掉了，形成了一个死循环，那么需要将其中onselectclass（）改为：onselectclass1（）

（2017/10/21）jrebel激活破解

具体参照[http://www.cnblogs.com/wang1024/p/72111941.html](http://www.cnblogs.com/wang1024/p/7211194.html)

**①准备工作：*JRebel6.4.3***插件的[**百度网盘下载**](http://pan.baidu.com/s/1o89efJ8)密码：o8c6  文件中包含插件的离线安装压缩包（**jr-ide-intellij-6.4.3\_13-16.zip**）和破解文件（**破解文件不支持其他版本的JRebel插件破解**）

　附JRebel插件官网下载地址（多版本）：[https://plugins.jetbrains.com/idea/plugin/4441-JRebel-for-intellij](https://plugins.jetbrains.com/idea/plugin/4441-jrebel-for-intellij)

②　　**注意：***已安装JRebel插件的需要在settings中将插件更新到最新版后才能将其卸载（已经是最新版的可以直接卸载），在安装完破解版JRebel后就不能再更新插件，否则破解会失效。*

*安装完成后发现这个版本和最新版相比反应速度会慢很多，很多时候需要刷新2次更新才能生效*

**③ JRebel安装与破解：**

1、 安装：在idea中点击file->settings，再点击plugins->install plugin from disk，选择JRebel插件的离线安装文件，点击确定后重启idea。

2、 破解：关闭idea，打开压缩包中的破解文件夹，替换一个licence文件与两个jar包 ​：

　　1) 将本机 C:\Users\你的用户名\.JRebel\JRebel.lic替换为下载的补丁包里的 JRebel.lic；

　　2) C:\Users\你的用户名\.IntelliJIdea2017.1\config\plugins\jr-ide-idea\lib\JRebel6\JRebel.jar

　　　 3) C:\Users\你的用户名\.IntelliJIdea2017.1\config\plugins\jr-ide-idea\lib\JRebel\JRebel.jar

④ 然后启动idea，在file->settings->plugins->JRebel中查看是否已显示激活，激活后显示valid，图标为绿色（如果没激活的话，就去试试翻qiang注册正版吧）点击activate其中有三种激活方法选择其中第二个。ZYJ

⑤ 激活后，设置JRebel，选择settings -> JRebel -> advanced 选择Jrebel 6 Agent，然后重启idea。Jrebel的其他配置按默认就好成功了!

（2017/10/22）

1.在这里需要注意@ManyToOne和@OneTwoOne的区别！！！！！！！



1. 0的阶乘就是1,
2. final的值是不能改变的  
   static是全局变量，只能有一个  
   static final int a的意思就是不管你new多少次只有一个a变量且是不可改变的

(2017/10/26)

**当通过new Scanner(System.in)创建一个Scanner，控制台会一直等待输入，直到敲回车键结束，把所输入的内容传给Scanner，作为扫描对象。如果要获取输入的内容，则只需要调用Scanner的nextLine()方法即可**

**System.out.println**

**Scanner s=new Scanner(System.in);**

public class TestScanner {   
        public static void main(String[] args) {   
                Scanner s = new Scanner(System.in);   
                System.out.println("请输入字符串：");   
                while (true) {   
                        String line = s.nextLine();   
                        if (line.equals("exit")) break;   
                        System.out.println(">>>" + line);   
                }   
        }   
}

(2017/10/27)

扫描从控制台输入的字符,Scanner是用于扫描输入文本的类名，input仅仅是对象名，你可以换成任意的非java保留着，如a、b、c都行，new是给对象new（开辟）一个空间存储扫描进来的东西，System.in 是PrintStream（字节打印流）的一个实例 ，只能按字节读取

Scanner input=new Scanner(System.in);

Int N=input.nextInt();

System.out.println(N>90?”A”:(N>60?”B”:”C”))

(2017/10/28)

1. char c=a.charAt(0);

获取字符串a的第一个字符，将其赋值给字符变量c。

（2017/11/3）

Spring

今天开始学习spring

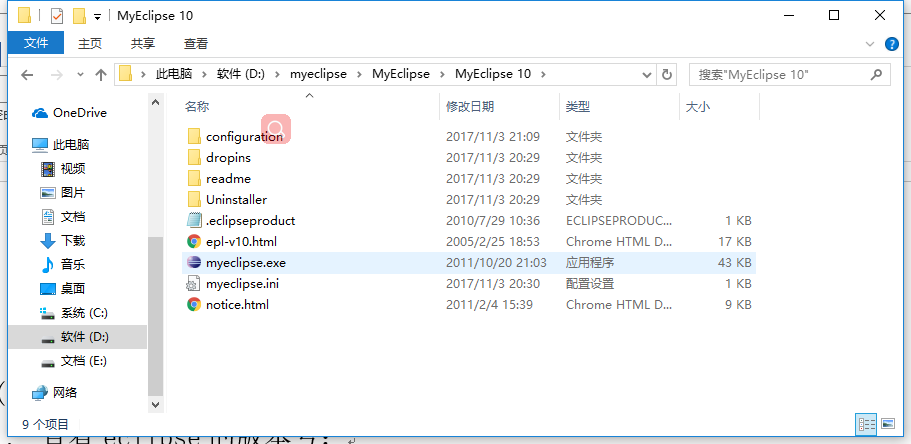
1. spring是以个IOC（DI）和AOP（面向切面编程）容器框架，IOC是反转控制,DI是依赖注入，是IOC的另一种表达方式
2. 描述Spring:
   1. 轻量级：Spring是非侵入性的
   2. 容器
   3. 一站式容器
3. 安装Myeclipse
   1. 百度搜索myeclipse下载和安装破解教程



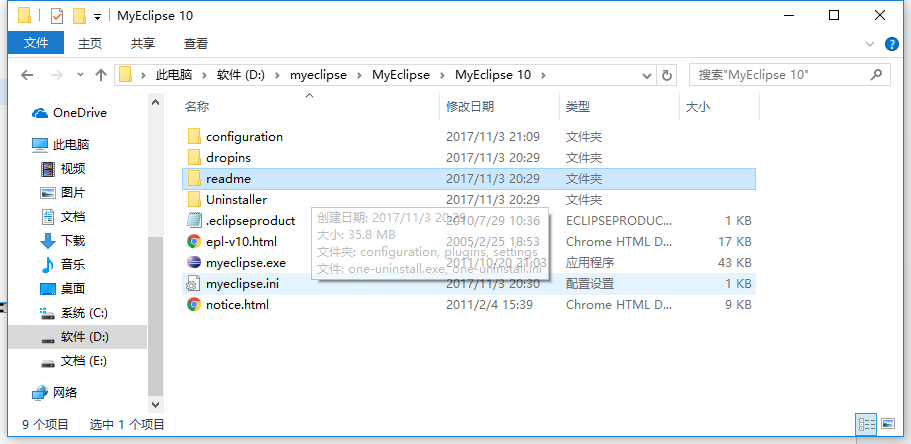
* 1. <http://jingyan.baidu.com/article/e9fb46e1752f797520f7665d.html>
  2. MyEclipse安装包与破解文件（下载地址：<http://pan.baidu.com/s/1qWwQEGc>）
  3. 跟着步骤走ok了

(2017/11/4)

1. 查看eclipse的版本号,两种方法：
2. 找到Eclipse的解压目录即Eclipse.exe所在目录.eclipseproduct文件



1. 目录里边有一个readme打开，里边有一个readme\_eclipse.html



1. 下载eclipse2017时间有点漫长，打算用idea创建一个简单的pring项目但跟着步骤，<http://blog.csdn.net/cflys/article/details/70598903>

找不到spring-config.xml文件

解决方法：导入spring-context的依赖

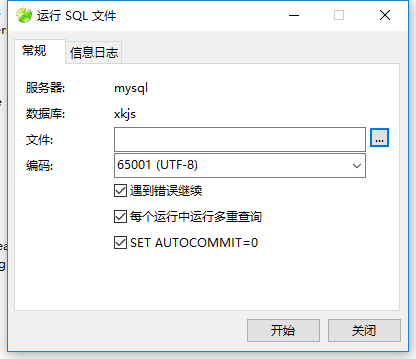
右击new——XML Configuration File—— Spring Config——ok,解决了

1. Spring到底帮我们做了什么，当我们没有使用Spring的时候，调用sayHello()方法需要三步：   
   . 创建一个HelloWorld的实例对象   
   . 设置实例对象的name属性   
   . 调用对象的sayHello()方法   
   后来也需要3步：   
   . 创建一个Spring的IOC容器对象   
   . 从IOC容器中获取Bean实例   
   . 调用sayHello()方法   
   然后我们探究了什么时候new的helloWorld对象，我们可以看出，Spring帮我们完成了前2步，也就是创建实例对象以及设置对象的属性，也就是说我们可以把对象的创建和管理工作交给Spring去完成，不需要自己去new对象，也不要去设置对象的属性，只要写好Spring的配置文件，Spring就可以帮我们去做，当我们需要对象的时候，直接去找Spring去要就行。
2. 配置bean
   1. Class :bean的全类名，通过反射的方式在IOC容器中创建Bean，所以要求Bean中必须有无参数的构造器
   2. Id：通过id来获取bean
   3. ApplicationContext代表IOC容器，事实上它是一个接口→ConfigurableApplicationContext（扩展与ApplicationContext）底下有两个实现类→(ClassPathXmlApplicationContext 该实现类从类路径下来加载配置文件)
   4. 用ApplicationContext读取bean的方法：直接用id进行读取。

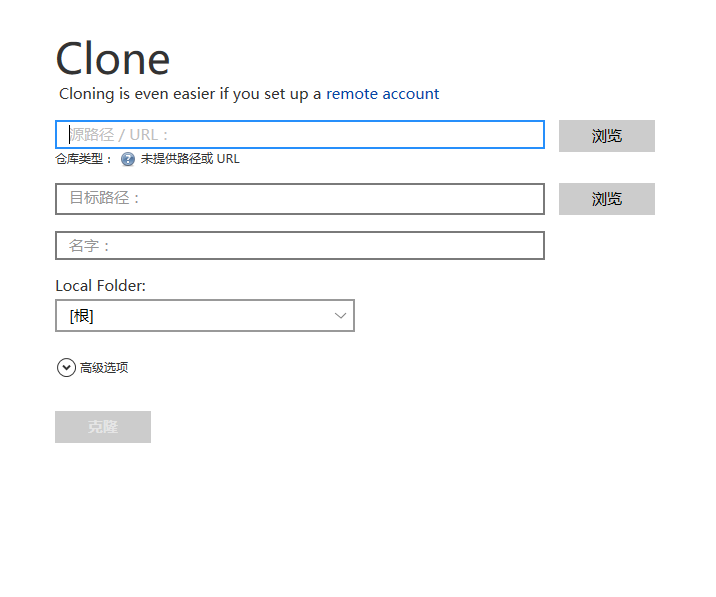
（2017/11/30）

一、学科竞赛项目导入带有一部分数据的导入

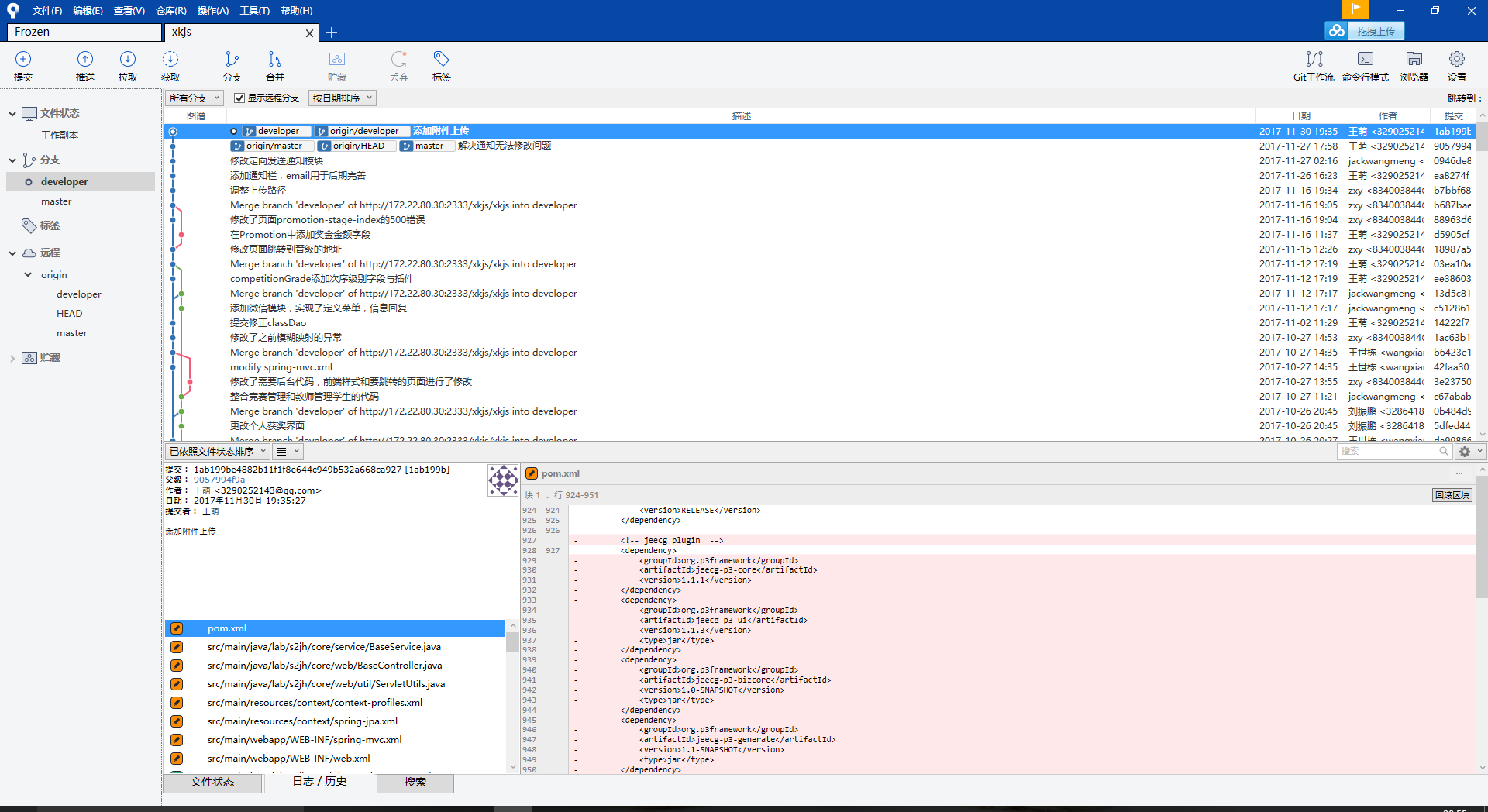
1.导入一组数据 mysql里边 右击项目xkjs 运行SQL文件，选择你那个.sql文件开始→关闭→刷新→ok



2.sourceTree里边导入项目 先登录自己实验室账号http://172.22.80.30:2333→选择你的项目点击复制→新建/克隆



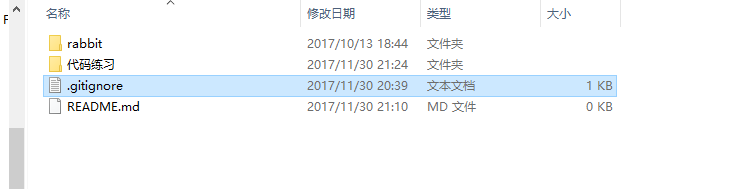
→http://Frozen@gitlab/xkjs/xkjs.git→将中间的红色字体改为172.22.80.30:2333→（自动生成啦）点击目标路径 浏览选择自己在E盘创建的xkjs文件夹（文件夹里边必须是空的）→确定→检出分支（远程里边origin→developer→右击检出origin/developer分支）

3.打开idea 重新导入

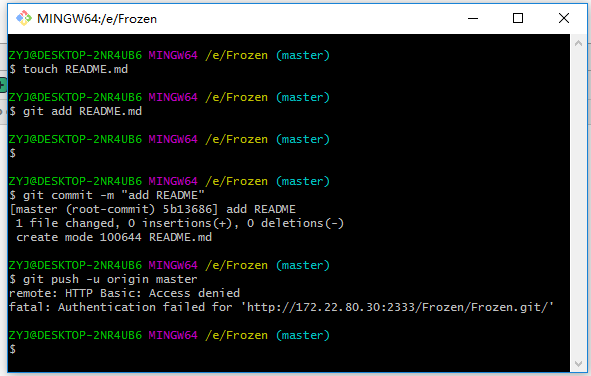
二．自己创建一个项目使之在不同的地方都可以继续写自己的代码。

1.同上创建一个自己的project：Frozen，创建文件夹

2. 复制这个.gitignore文件



3.创建一个master



4.只需要将自己的文件比如rabbit（之前写的算法）文件夹直接复制到 Frozen底下即可

5.项目导入之后，直接推送到服务器。就可以在gitlab上边看到你提交上的东西哦。

三．安装spring插件在eclipse上

<https://www.cnblogs.com/MrYoodb/p/7574566.html>

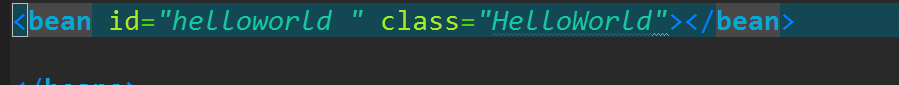
springsource-tool-suite-3.9.1.RELEASE-e4.6.3-updatesite.zip离线安装步骤

（2017/12/1）

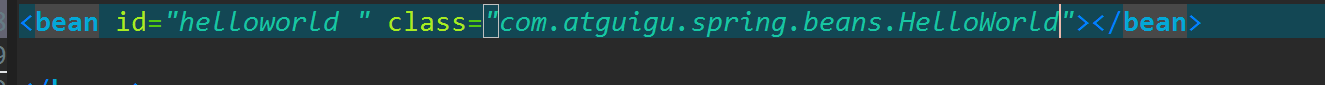
1. <http://blog.csdn.net/waeceo/article/details/50935506>

eclipse配置spring(最新版本或者是以前的版本)右击一个绿色的spring

1. 找到myeclipse的spring配置文件右击项目选择了一个绿色的什么install回宿舍可以试一下，咳咳
2. 自动补全捏哦也alt+/（代码辅助）



将光标放在红线上边，摁快捷键



Ok了。

4.快速修复”Ctrl+1”快速修复

（2017/12/02）

1.今天莫名其妙JDK没了，重新安装回去了

JDK安装与环境变量配置

<https://jingyan.baidu.com/article/6dad5075d1dc40a123e36ea3.html>

2.创建了一个java-web 通过谷歌

<http://localhost:8080/HelloWorld>

就可以出来啦

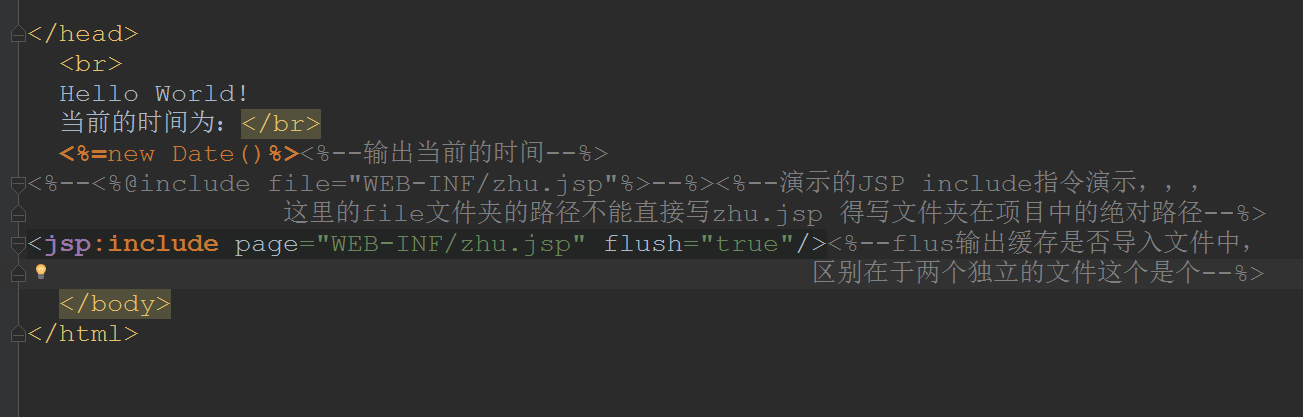
3.直接将eclipse的工作环境放到SourceTree里边的Frozen里边

（2017/12/03）、

1.<http://blog.csdn.net/memgxingfeixiang/article/details/52486760>

gitlab与sourcetree的混合使用

2.



3.随机字符的生成验证

<http://bbs.csdn.net/topics/230071985>

(2017/12/4)

1. 直接访问WEB-INF目录下的JSP页面的方法（controller）

（2017/12/06）

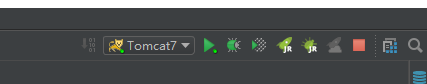
1. 安装SourceTree

<http://www.jianshu.com/p/3478e2a214a1、>

1. 注意用springMvc写一个简单的HelloWorld需要jia

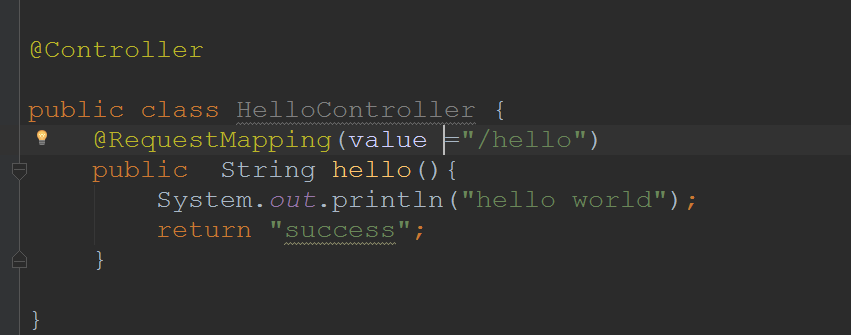
<https://www.cnblogs.com/goodcheap/p/6518931.html>

需要点一下这个将jar包引进去否则会出错捏。



1. mvc框架里边一般有model(里边一般为基本的java bean)，view，controller
2. 创建一个java package

正常写一个HelloWorld.java类正常写一个方法，



需要添加配置自动扫描的

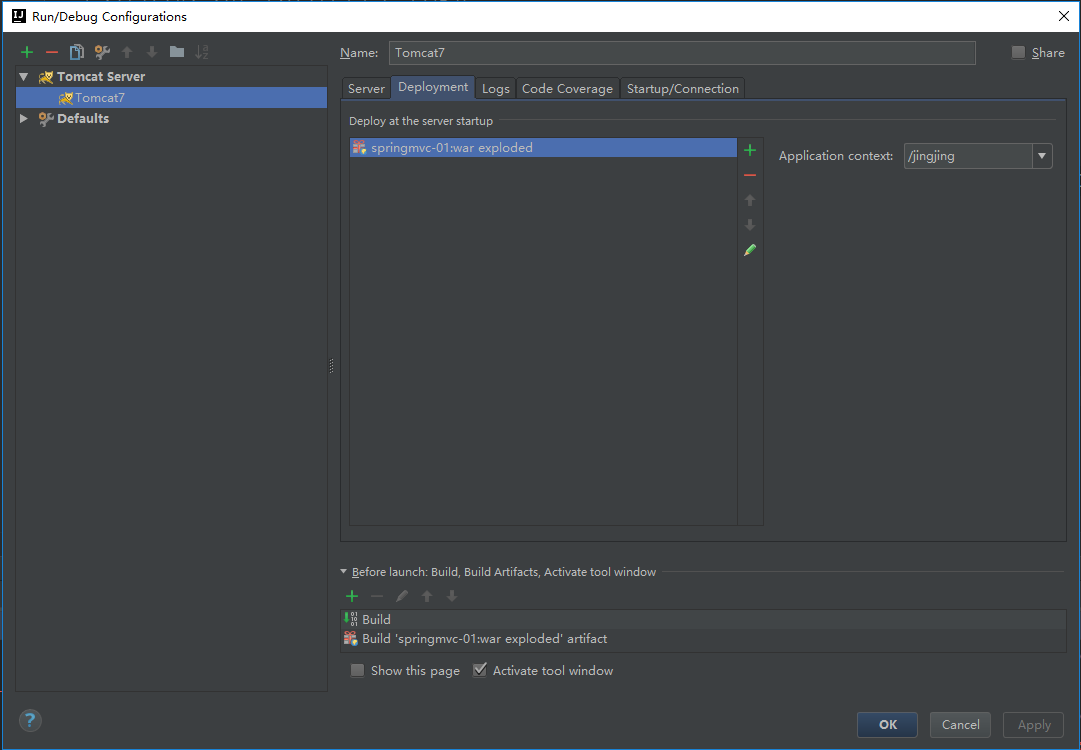
(2017/12/07)

1. <http://blog.csdn.net/industriously/article/details/52851588>

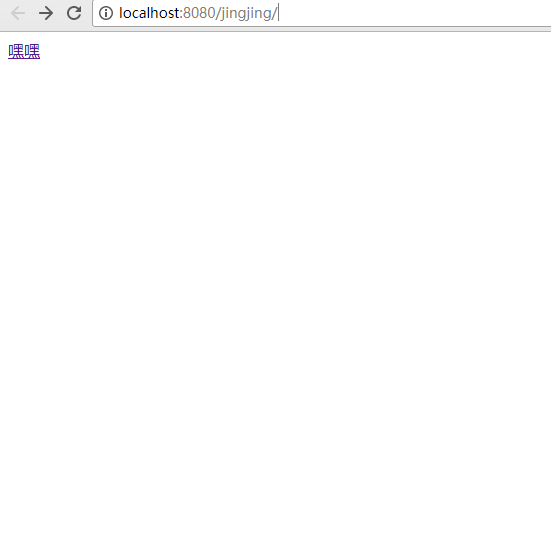
使用IntelliJ IDEA开发Spring MVC HelloWorld

1. Directory目录建立一个views目录记得呢！

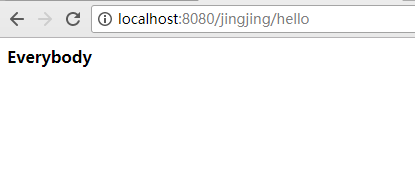
3.改变途中圈住的那个：



由之前的 <http://localhost:8080> 变为<http://localhost:8080/jingjing/>



<http://localhost:8080/hello>变为<http://localhost:8080/jingjing/hello>



1. RequestMapping不仅可以修饰方法还可以修饰类

(2017/12/12)

1. <https://www.cnblogs.com/chiangchou/p/mappedBy.html>[**hibernate基于注解的维护权反转：@OneToMany(mappedBy=)**](http://www.cnblogs.com/chiangchou/p/mappedBy.html)
2. <http://zmfkplj.iteye.com/blog/319990>

**这样写会只成生成表A和表B，B中会有一个到表A的外键。但是如果不加mappedBy=”a”， 那么就会再生成一张A\_B表。**

(2017/12/16)

一、HQL数据查询基础

1. 检索对象持久化对象——from子句
2. 选择查询对象指定属性——select子句
3. 限制对象的范围——where子句
4. 排序——查询结果来排序————order by

二、HQL的定义

1.HQL是 面向对象的查询语言

HQL-------映射配置的持久化类及其属性

SQL----------数据库表

三、Query接口

1.Query接口定义执行查询的方法

HQL语句通过hibenate框架负责解析HQL语句和根据映射配置信息→SQL语句对数据库进行相应操作（完成这一操作利用的就是Query接口）

2.方法链编程：调用方法后返回方法结果的还是这个方法的对象（我们调用方法后直接调用该对象的其他方法），例如查询参数的动态设置，特别多个参数的设置

3.创建Query实例对象

Session接口的 createQuery()方法创建Query示例；

createQuery方法包含一个HQL语参数，createQuery(hql)

4.Query执行查询

Query接口的list（）方法执行HQL查询；

list()方法返回的数据类型为java.Util.List,List集合中存放符合查询条件的持久化对象

四、查询对象——from子句

1.from子句（不可或缺）

HQL只要from子句，hibenate框架自动进行解析； SQL中不仅需要from还需要select子句查询的字段信息

2.from子句中持久化类的引用

不需要全限定名 com.imooc.Seeler 直接使用持久化类名Seller

String hql =”from Seller ”;

这是因为auto-impor（自动引入）缺省情况

3.使用别名

通常为被查询的类指定别名；在HQL语句其他语句通过别名引用该类；

String hql =”from Seller as seller ”;//别名

String hql =”from Seller seller ”;//其中as可以省略

String hql =”from Seller s, Customer c”;//你也可以把它变成单字母;用逗号隔开查询持久化类

五、选择——select子句以哪种形式返回——提高程序的高效性指定的查询

1.指定选择的对象及属性后以哪种数据类型（默认）——对象数组

String hql =”select s.name,s.tel,s.address,s.star from Seller s”;//只需要这么几个信息则用到了select子句

Query query=session.createQuery(hql);

List<Object[]> list=query.list();通过对象数组来返回的，接收一下

for(Object[] objs:list){ //验证一下输出

System.out.println(“name”+objs[0]);//位置和索引顺序是一样的上边

System.out.println(“tel”+objs[1]);

System.out.println(“address”+objs[2]);

System.out.println(“star”+objs[3]);

}

**当返回一个属性时，返回的数据类型就不是我们的对象数组，而是对象！注意一下这是一个特殊的实例！！！！**

String hql=”select s.name from Seller s”;

Query query=session.createQuery(hql);

List<Object> list=query.list();//指定的是对象类型

for(Object obj:list){

System.out.println(“name”+obj);

}

2.list集合

Select子句中使用new list指定我的返回查询结果。

String hql =”select new list( s.name,s.tel,s.address) from Seller s”;

Query query=session.createQuery(hql);

List<List> lists=query.list();//执行查询查询结果List集合形势

for(List list:lists){ //循环验证;互相对应！！！咳咳中间 那个表示底下输出System.out.println()里边的map.get(“0”)这个

}

3.map 自定义

使用new map来指定返回的是map类型

String hql=”select new map(s.name,s.tel,s.address )from Seller s”;//先from，select最后new map

//定义query对象

List<Map> maps=query.list();//执行查询，查询结果是map形式

for(Map map:maps){

System.out.println(“name:”+map.get(“0”));

System.out.println(“tel:”+map.get(“1”));

System.out.println(“address:”+map.get(“2”));

}

注意啦注意啦每个属性都设置一个别名,获取属性信息更方便理解

String hql=”select new map(s.name as name,s.tel as tel,s.address as address) from Seller s”;

System.out.println(“name:”+map.get(“name”));

System.out.println(“tel:”+map.get(“tel”));

System.out.println(“address:”+map.get(“address”));

4．自定义构造器

首先我们需要在持久化类中定义对应的构造器（参数对应属性信息），select语句调用我们定义的构造器

商家类 Seller entity里边

public Seller(String name,String tel,String address){

this.name=name;

this.tel=tel;

this.address=address;

}

定义一个测试方法。。一直可以定义新的测试方法

String hql=”select new Seller( s.name,s.tel,s.address) from Seller s”;

Query query=session.createQuery(hql);

List<Seller> sellers =query.list();

for(Seller seller:sellers){

System.out.println(“name:”+seller.get(“0”));

System.out.println(“tel:”+ seller.get(“1”));

System.out.println(“address:”+ seller.get(“2”));

}

注意默认构造器的补充！！！！这样就不会发生异常

public Seller（）{

}

5. 获取独特的查询结果——distinct关键字

用distinct关键字来消除查询中重复的元素例如性别：男、女。

public class CustomerTest{

@Test

Public void testDistinct(){

String hql=”select **distinct** c.sex from Customer c”;

Query query=session.createQuery(hql);

List<Object> list=query.list();

for(Object obj:list){

System.out.println(“sex:”+obj);

}

}

}

本来结果为男女男消除了重复的元素

六、限制where子句

1.比较运算

= 相等

<>不等

<,>

null值判断——is [not]null

hibernate通过解析

X=null——x is null

X<>null——x is not null

例子

CommodityTest类里边定义一个测试方法

Public void testWhere1(){

String hql=”from Commodity c where c.price>400”;//hql语句其中价格大于400的

String hql=”from Commodity c where c.description=null”;//sql语句中=null是不行的但在hql中这种写法是可以的hibernate框架将其解析，<>不等于也可以

Query query =session createQuery(hql);//定义query对象

List <Commodity> commodities=Query.list();//执行查询，获取查询的结果

for(Commeodity c:commodities){ //验证查询结果

System.out.println(“name:”+c.getName());

}

}

2.范围运算

[not] In(列表)

[not] between 值A and 值B

CustomerTest类里边写一个测试类

{

String hql=”from Customer c where c.age in(20,40)”;

String hql=”from Customer c where c.age not in(20,40)”;范围取反

String hql=”from Customer c where c.age between 20 and 40”;

String hql=”from Customer c where c.age not between 20 and 40”; 取反

Query query=session.createQuery(hql);

List<Customer> customers=query.list();

for(Costomer c: customers){

System.out.println(“name:”+c.getName());

System.out.println(“age:”+c.getAge());

}

}

3.字符串模式匹配

like关键字

通配符： %：任意个字符

\_ :一个字符

CustomerTest类

public void testWhere3(){

String hql=”from Customer c where c.name like ‘张\_’”;//寻找姓张的单引号

String hql=”from Customer c where c.name like ‘%北京%’”;含有北京的

Query query=session.createQuery(hql);

List<Customer> customers=query.list();

for(Customer c:customers){

System.out.println(“name:”+c.getName());

System.out.println(“address:”+c.getAddress());

}

}

4.逻辑运算

CommodityTest类

public void testWhere2(){

String hql=”from Commodity c where c.price between 100 and 5000 and c.category like ‘%电脑%’”;

String hql=”from Commodity c where c.price between 100 and 5000 or c.category like ‘%电脑%’”;

Query query=session createQuery(hql);

List<Commodity> commodities=query.list();

for(Commodity c:commodities){

System.out.println(“name:”+c.getName());

System.out.println(“category:”+c.getCategory());

System.out.println(“price:”+c.getPrice());

}

}

5.集合运算

is [not] empty 集合【不】为空，不包含任何元素————转化为sql中exists

member of 元素属于集合———转换为in

OrderTest类里边

{

public void testWhere1(){

String hql=”from Order o where o.orderItems is not empty”;

Query query =session.createQuery(hql);

List<Order> orders=query.list();

for(Order o:orders){

System.out.println(“name:”+ o.getCustomer().getName());

System.out.println(“amount:”+o.getAmount());

}

}

}

6.HQL中使用＋－×÷运算符

可以在where \select中都可以使用

commodityTest类里边创建测试类

public void testwhere1(){

String hql=”from Commodity c where c.price\*5>3000”;}//五件商品总价大于3000

Query query=session.createQuery(hql);

List<Commodity> commodities=query.list();

for(Commodity c:commodities){

System.out.println(“name:”+c.getName());

System.out.println(“price:”+c.getPrice ()\*5);//注意要乘5

}

7.查询单个对象（uniqueResult方法）

CunstomerTest类里边

public void testWhere4(){

String hql=”from Customer c where c.name=’张三’”;

Query query=session.createQuery(hql);

Customer c=(Customer)query.uniqueResult();

System.out.println(“name:”+c.getName());//这儿是唯一的so不用循环输出

}

七、Order By子句

默认 升序 asc

降序排序Desc

在CommodityTest类里边

Public void testWhere3(){

String hql=”from Commodity c order by price asc”;//注意order by

String hql=”from Commodity c order by Seller.id asc,price desc,name asc”;//要用逗号隔开，从前往后排

Query query=session.createQuery(hql);

List<Commodity> commodities=query.list();

for(Commodity c:commodities){

System.out.println(“name；”+c.getName());

System.out.println(“sellerId：”+c.getSeller.getId());

System.out.println(“sellerName:”+c.getSeller.getName());

System.out.println(“price”+c.getPrice());

}

}

 select \* from a inner join b on a.id=b.id这两个SQL语句，是不是都是先生成表a和表b的笛卡尔积之后，再按照a.id=b.id 的条件 从笛卡尔积 中 提取符合a.id=b.id 的记录

（2017/12/19）曾今迷失了我自己，含泪默默哭泣

1.entity开始看

[@MetaData](mailto:1.@MetaData)元数据（value属性值，comments注释）

@OneToMany(mappedBy=”college”,cascade=CascadeType.All) //作为外键，学院关联竞赛申请，cascade（级联）级联在编写触发器时经常用到，触发器的作用是当 主控表信息改变时，用来保证其关联表中数据同步更新。若对触发器来修改或删除关联表相记录，必须要删除对应的关联表信息，否则，会存有脏数据。所以，适当的做法是，删除主表的同时，关联表的信息也要同时删除，在hibernate中，只需设置cascade属性值即可。@JoinColumn(name=“teache\_id”,nullable=false)//关联教师表，这个字段保存时必须有值

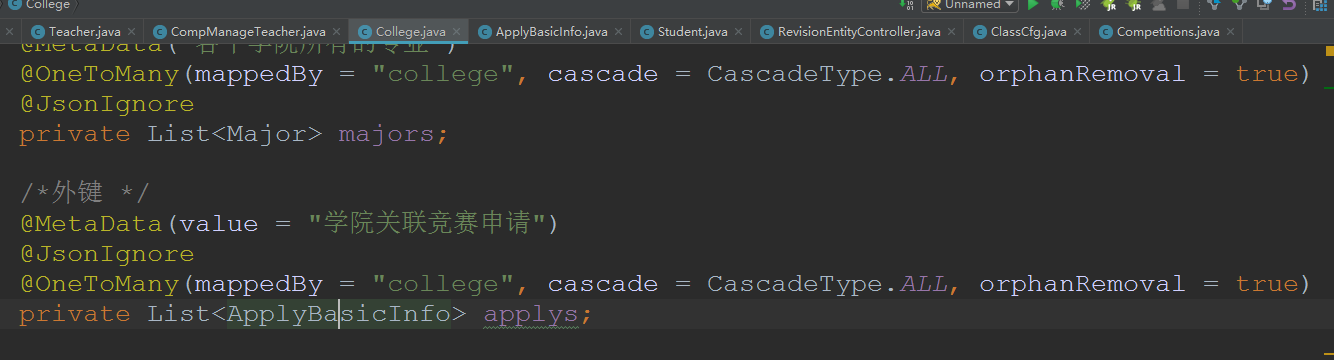
@JsonIgnore//一般这两句是一起用的这个时候从后台推数据到前台的时候,就会把teacher这个引用属性给忽略掉。作用是json序列化时将java bean中的一些属性忽略掉,序列化和反序列化都受影

2.ClassCfg实体里边写到了

学院的属性则

@ManyToOne(fetch=FetchType.LAZY)//很多班级对应一个学院！！！

3.学院实体里边写一个外键 关联竞赛申请的



(2017/12/25)

1. ?表示一个字符

\*表示任意一个字符

\*\*匹配多层路径

(2017/12/26)

1.REST（资源表现层状态转化）

GET获取 例如： /order/1 得到id=1的order订单

POST新建 /order新增order

通常有这两种请求，springmvc通过HiddenHttpMethodFilter(需要在xml中配置一下)

PUT更新 /order/1更新id=1的订单

DELETE删除 /order/1 删除id=1的订单

(2018/1/12)

1. 页面中调用函数--之${fn：}内置函数

fn:length 获取字符串的长度 ，或者数组的大小 ${fn:length(shoppingCart.products)}

<http://blog.csdn.net/sdta25196/article/details/78999654>

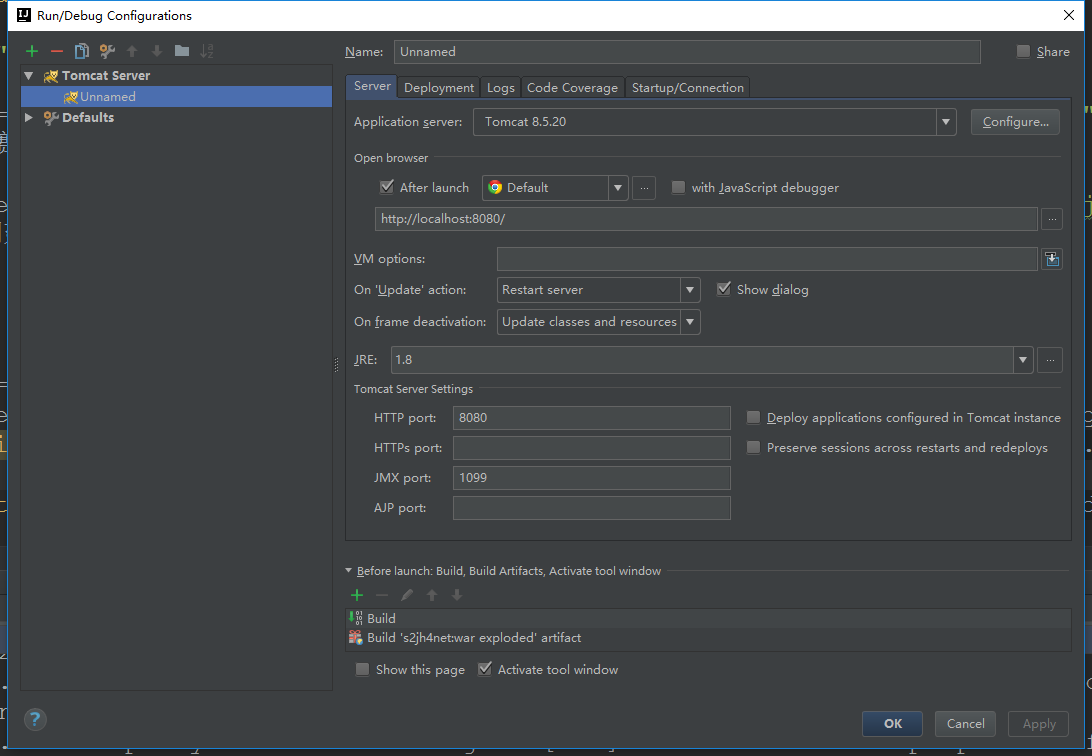
1. readonly=”readonly”

把输入字段视为只读，只读字段是不能修改的。不过，用户仍然可以使用 tab 键切换到该字段，还可以选中或拷贝其文本。

例子：

Country:<input type=”text” name=”country” value=”China” readonly=”readonly”/>

1. 热部署jre+Update Classes and resources才可以用jrebel热部署。



(2018/1/13)

* + 1. <http://blog.csdn.net/zhanjianshinian/article/details/45030911>分页
    2. 尤其是在制作导航时，float(浮动)自适应居中是经常运用得到，无论使用

text-align: center（文本居中），

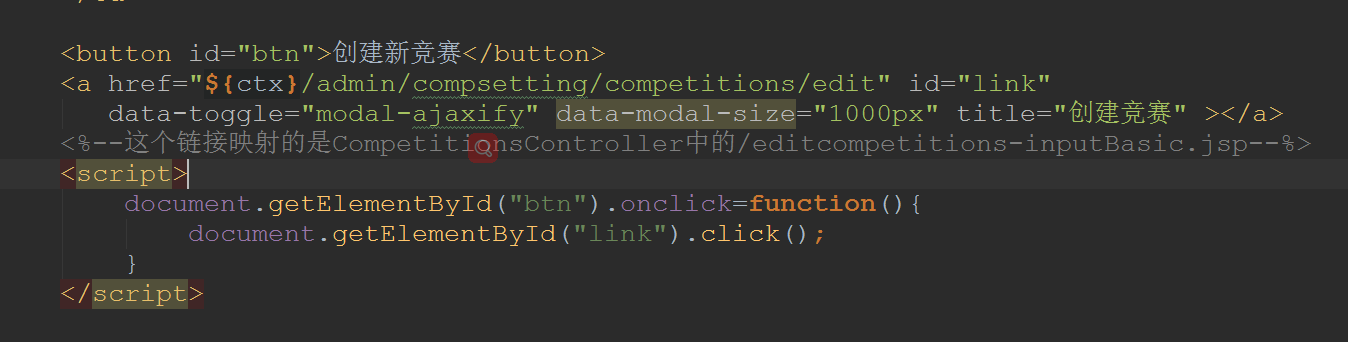
vertical-align: middle（中线对齐）

不能解决

<div style=”float=right”>比赛和安排</div>//向右浮动对齐

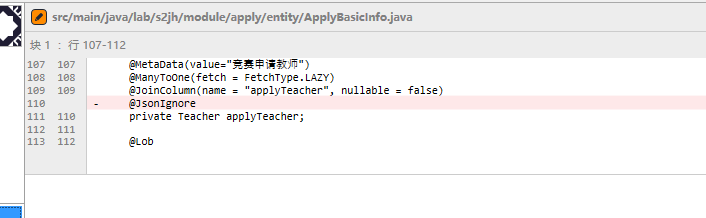
 3.将<a></a>链接实现按钮形式<button></button>

<button id="btn"></button>  
<a href="https://www.baidu.com/" id="link">link</a>//注意将href这儿写成自己需要的链接   
<script>  
document.getElementById("btn").onclick=function(){  
　　document.getElementById("link").click();  
}  
</script>

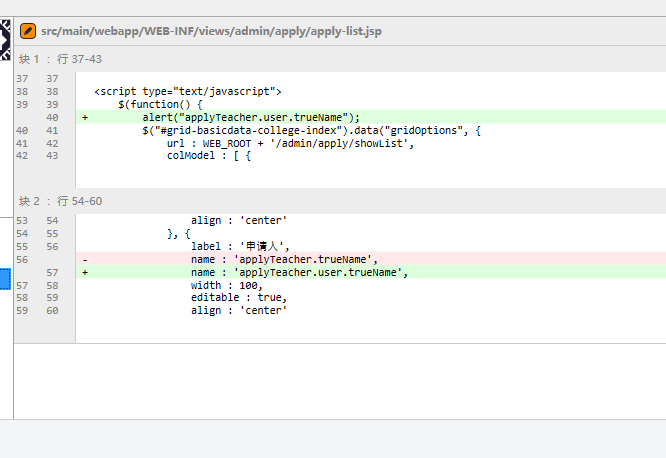


（2018/1/14）

1.Json中传送数据，将数据@JsonIgnore过滤掉，不传送，故在这把竞赛申请老师的类里边@JsonIgnore删掉就ok



这儿也不对 Teacher applyTeacher.trueName//这儿的TrueName是User实体的不可以所以得写成applyTeacher.user.trueName



2.Jackson相关:

使用Jackson相关的注解时一定要注意自己定义的属性命名是否规范。 命名不规范时会失去效果。（例如Ename ，Eage 为不规范命名。“nameE”，“ageE”为规范命名）我在此处掉坑半个小时。至于命名规范，大家自己搜索。其实不要太奇葩的命名都是可以得。如果使用@JsonIgnore注解不起效时请注意一下你的属性名字是否规范

①@JsonIgnoreProperties

此注解是类注解，作用是json序列化时将java bean中的一些属性忽略掉，序列化和反序列化都受影响。

写法将此标签加在model 类的类名上 ，可以多个属性也可以单个属性

//生成json时将name和age属性过滤

@JsonIgnoreProperties({"name"},{"age"})

public class user { private String name; private int age; }

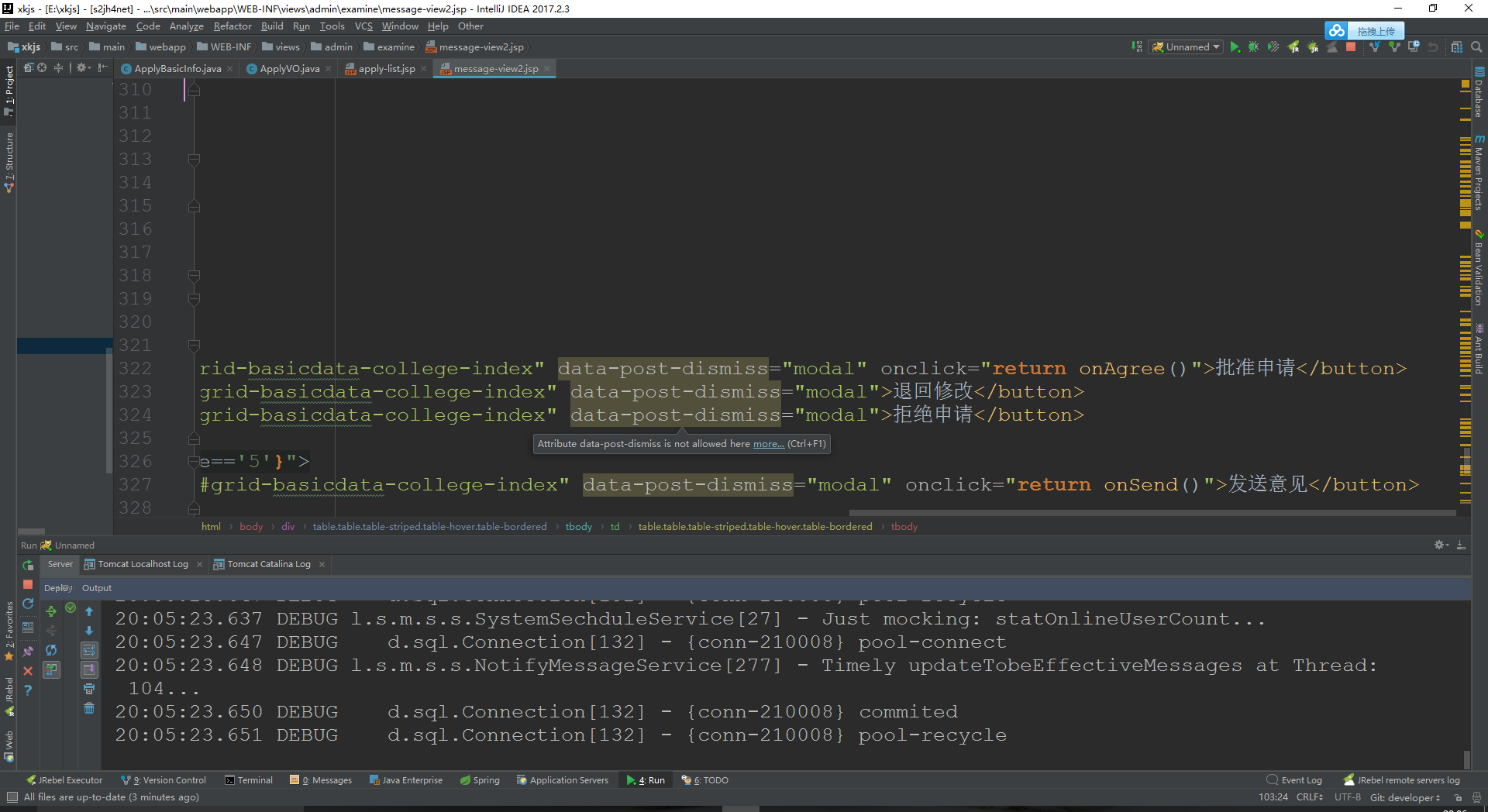
②@JsonIgnore

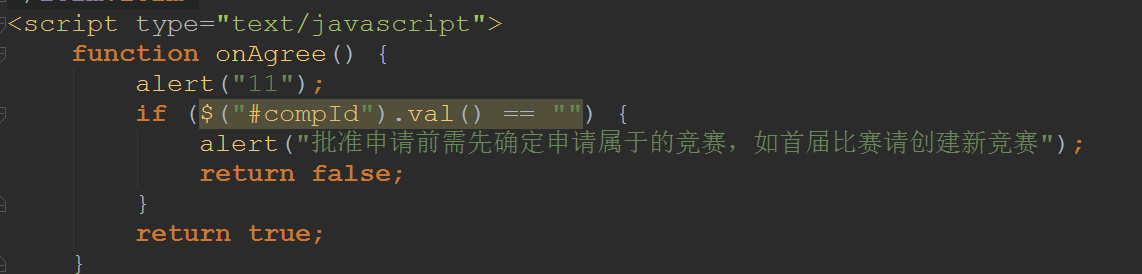
此注解用于属性或者方法上（最好是属性上），作用和上面的@JsonIgnoreProperties一样。

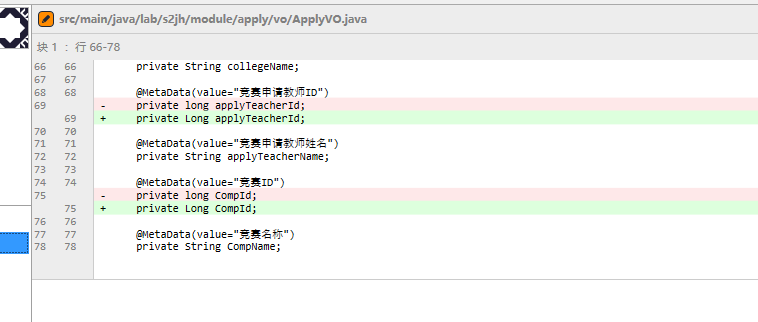
生成json 时不生成age 属性   
public class user {   
private String name;   
@JsonIgnore   
private int age;   
}

<http://localhost:8080/admin/apply/showList>

1. 注意这儿（大写Long有空值）Long有null值，long小写没有空值，而是0.







（2018/1/15）

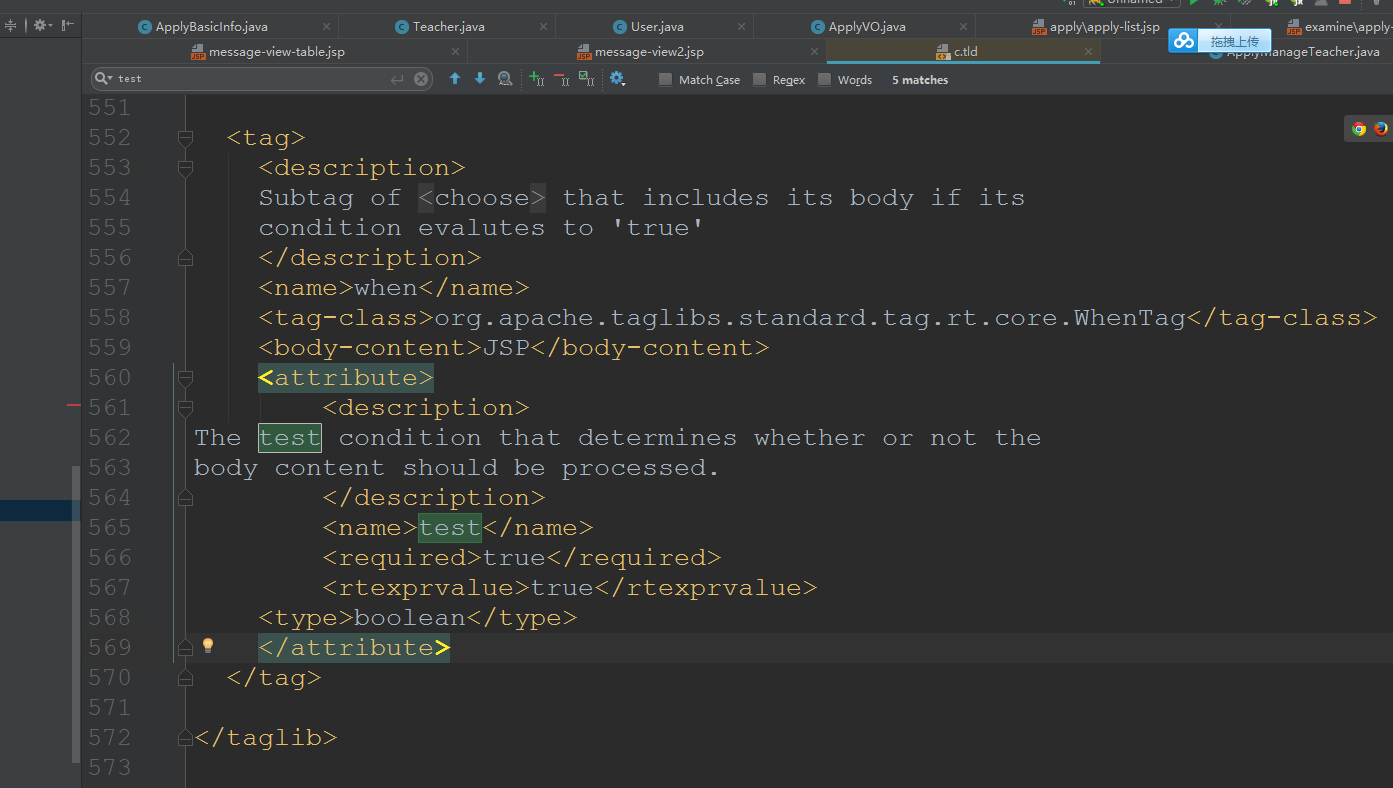
1.自定义标签tag

tld文件，配置我们自定义的标签-------

<short-name>th</short-name>标签中的th代表我们标签缩写

<uri></uri>标签是我们链接JSP的桥梁

<name>showip</name>标签名



其中，<required>yes</required><!-- 表示该属性，一定要使用 -->

<rtexprvalue>true</rtexprvalue><!-- 属性是否能用表达式 -->

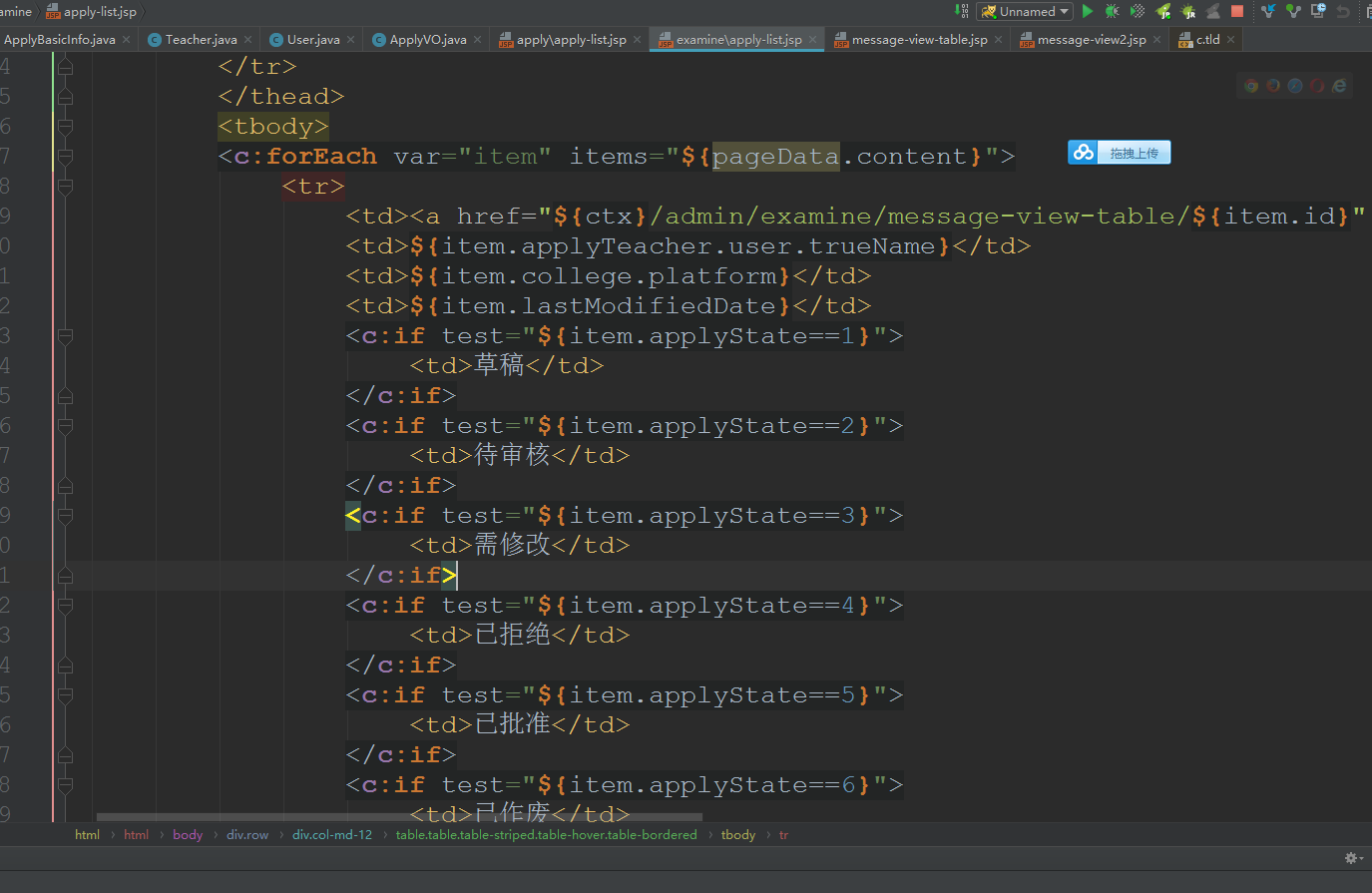
设置tld文件成功，新建一个JSP文件，设置头文件**<%@taglib uri="" 填写我们tld中的uri prefix=""%>填写标签缩写比如上文中的th**

接着在body标签中填写我们的自定义标签**<th:showIp></th:showIp>缩写标签+标签名**

<http://blog.csdn.net/ethansky/article/details/3607465>

自定义标签的 使用

2.test="",引号里面结果为boolean类型，可以跟普通代码里面一样，用用与或非运算，只要引号里面运算结果是boolean类型即可；<br>"或"运算可以用"||"和"or"表示，"与"运算可以用"&&"和"and"表示。



# 3. [bootstrap modal的data-dismiss属性](http://www.cnblogs.com/flying607/p/5962818.html)

<button type="button" class="btn default" data-dismiss="modal">关闭</button>

如果你的modal弹窗里面加上这个按钮，那么点击则会关闭当前弹窗，关键在于data-dismiss="modal"，它让按钮有了这个功能。这是bootstrap.js中MODAL插件相关代码中定义的:

**this.$element.on('click.dismiss.bs.modal', '[data-dismiss="modal"]', $.proxy(this.hide, this))**

（2018/1/16）

1. <http://localhost:8080/admin/apply/showList>

通过这个可以看看有没有提交这个字段

（2018/1/17）

1. timeStamp 事件属性可返回一个时间戳。指示发生事件的日期和时间（从 epoch 开始的毫秒数）。epoch 是一个事件参考点。

2.个人申请中提交日期的显示

ApplyBasicInfo.setSubmitTime(new Date());

# 3JavaScript substring() 方法

*stringObject*.substring(*start*,*stop*)

//Start必须有，stop可有可无

<script type="text/javascript">

var str="Hello.。 world!"

document.write(str.substring(3,7))

</script>

输出：

lo w

### 返回值

一个新的字符串，该字符串值包含 *stringObject* 的一个子字符串，其内容是从 *start* 处到 *stop*-1 处的所有字符，其长度为 *stop*减 *start*。

（2018/1/18）

# 1.springmvc的ModelAttribute注解，在controller每个方法执行前被执行

2. 但一个是GET方法,一般用于检索.   
一个是POST方法,一般用于提交表单

（2018/1/20）

1. @Transactional注解没生效，导致需要事务回滚的时候，事务没有回滚，最终导致数据库中的数据产生了不统一。

2. @Column(nullable=false,unique=true)

unique=true是指这个字段的值在这张表里不能重复，所有记录值都要唯一，就像主键那样  
nullable=false是这个字段在保存时必需有值，不能还是null值就调用save去保存入库  
用法是不同的，不存在什么设置了一个就不需要另一个道理

3. Oracle之唯一性约束(UNIQUEConstraint)用法详解

<http://blog.csdn.net/baijianjun123456/article/details/51178399>

# 注意 ：1 目标

用示例演示如何创建、删除、禁用和使用唯一性约束。

# 2 什么是唯一性约束？

唯一性约束指表中一个字段或者多个字段联合起来能够唯一标识一条记录的约束。联合字段中，可以包含空值。

注：在Oracle中，唯一性约束最多可以有32列。

唯一性约束可以在创建表时或使用ALTER TABLE语句创建。

# 3 唯一性约束和主键的区别

主键（Primary Key）：所有组成主键的列都不能包含空值。唯一性约束（Unique Constraint）：如果唯一性约束由多列组成，其中的部分列可以包含空值。Oracle中不容许在相同列上既创建主键又创建唯一性约束。

# 4 创建表时定义唯一性约束

## 1）语法：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | CREATE TABLE table\_name  (      column1 datatype null/not null,    column2 datatype null/not null,    ...     CONSTRAINT constraint\_name UNIQUE (column1, column2,...,column\_n)  ); |

## 2）基于单列的唯一性约束示例：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | create table tb\_supplier  (    supplier\_id          number not null   ,supplier\_name        varchar2(50)   ,contact\_name         varchar2(50)   ,CONSTRAINT tb\_supplier\_u1 UNIQUE (supplier\_id)--创建表时创建唯一性约束  ); |

## 3）基于多列的唯一性约束示例：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | create table tb\_products  (    product\_id        number not null,    product\_name      number not null,    product\_type      varchar2(50),    supplier\_id       number,    CONSTRAINT tb\_products\_u1 UNIQUE (product\_id, product\_name) --定义复合唯一性约束  ); |

# 5 使用ALTER TABLE语法创建唯一性约束

## 1）语法

|  |  |
| --- | --- |
| 1  2  3 | ALTER TABLE table\_name  ADD CONSTRAINT constraint\_name  UNIQUE (column1, column2, ... , column\_n); |

## 2）示例准备，先创建表

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17 | drop table tb\_supplier;  drop table tb\_products;    create table tb\_supplier  (    supplier\_id          number not null   ,supplier\_name        varchar2(50)   ,contact\_name         varchar2(50)  );    create table tb\_products  (    product\_id        number not null,    product\_name      number not null,    product\_type      varchar2(50),    supplier\_id       number  ); |

## 3）基于单列的唯一性约束

|  |  |
| --- | --- |
| 1  2  3 | alter table tb\_supplier  add constraint  tb\_supplier\_u1  unique (supplier\_id); |

## 4）基于多列的唯一性约束

|  |  |
| --- | --- |
| 1  2  3 | alter table tb\_products  add constraint  tb\_products\_u1  unique (product\_id,product\_name); |

# 6 禁用唯一性约束

## 1）语法：

|  |  |
| --- | --- |
| 1  2 | ALTER TABLE table\_name  DISABLE CONSTRAINT constraint\_name; |

## 2）示例：

|  |  |
| --- | --- |
| 1  2 | ALTER TABLE tb\_supplier  DISABLE CONSTRAINT  tb\_supplier\_u1; |

# 7 使用唯一性约束

## 1）语法：

|  |  |
| --- | --- |
| 1  2 | ALTER TABLE table\_name  ENABLE CONSTRAINT constraint\_name; |

## 2）示例：

|  |  |
| --- | --- |
| 1  2 | ALTER TABLE tb\_supplier  ENABLE CONSTRAINT tb\_supplier\_u1; |

# 8 删除唯一性约束

## 1）语法：

|  |  |
| --- | --- |
| 1  2 | ALTER TABLE table\_name  DROP CONSTRAINT constraint\_name; |

## 2）示例：

|  |  |
| --- | --- |
| 1  2 | ALTER TABLE tb\_supplier DROP CONSTRAINT tb\_supplier\_u1;  ALTER TABLE tb\_products DROP CONSTRAINT tb\_products\_u |

(2018/01/21)

1. 后面缀上了!important的css语句，让浏览器首选执行这个语句，因为css有继承的样式，加上**!importanrt**可以覆盖父级的样式。

Style=‘background-color:#EEDFFF !important;‘//优先执行这个css属性

1. Display

<html><head>

<style type="text/css">

p {display: inline}

div {display: none}//隐藏了这段话

</style></head><body>

<p>本例中的样式表把段落元素设置为内联元素。</p>

<p>而 div 元素不会显示出来！</p>

<div>div 元素的内容不会显示出来！</div>

</body></html>

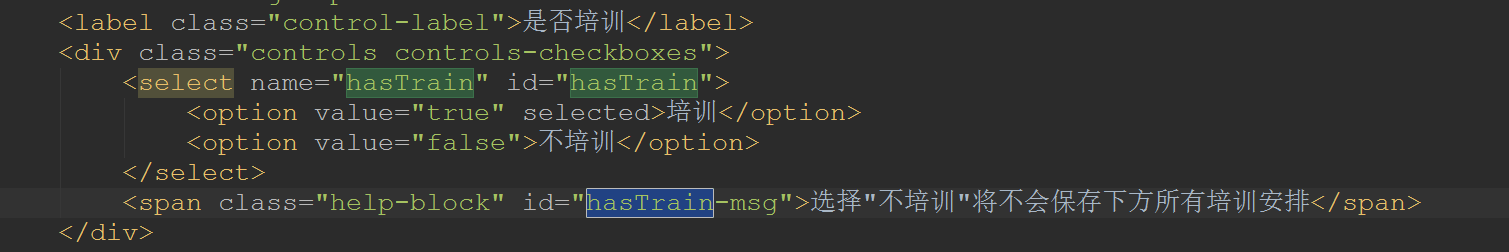
结果:

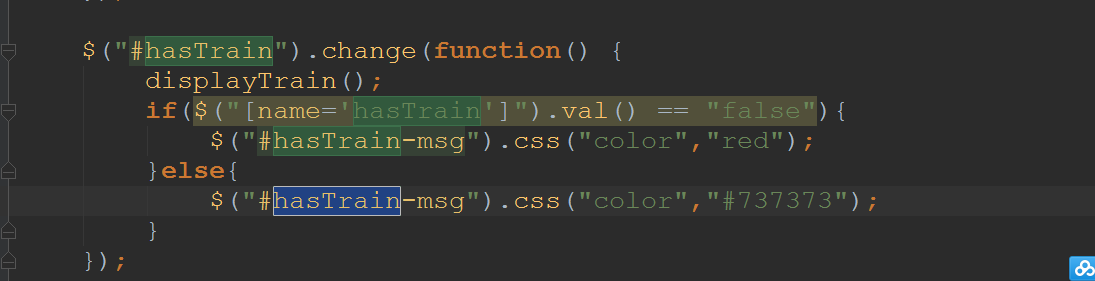
本例中的样式表把段落元素设置为内联元素。

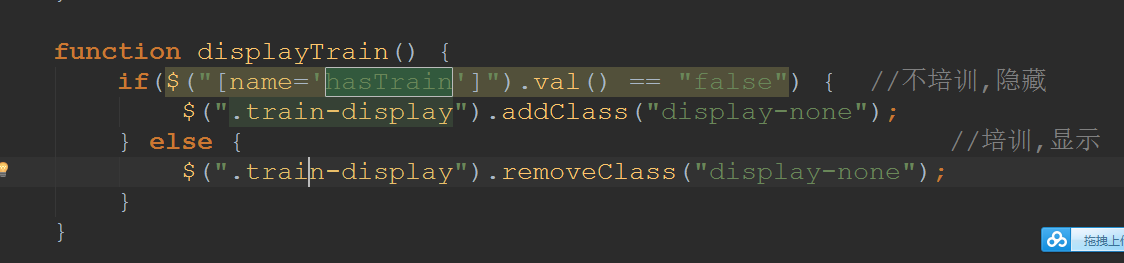
而 div 元素不会显示出来！

$('body').on('click', 'button[id="save\_add"]', **function** () {  
$(".deNew").css('display','none');  
});

3.







## 4. <c:forEach varStatus="status">中 varStatus的属性简介

我们常会用c标签来遍历需要的数据，为了方便使用，varStatus属性可以方便我们实现一些与行数相关的功能，如：奇数行、偶数行差异；最后一行特殊处理等等。先就varStatus属性常用参数总结下：

${status.index}      输出行号，从0开始。

${status.count}      输出行号，从1开始。

${status.current}   当前这次迭代的（集合中的）项

${status.first}  判断当前项是否为集合中的第一项，返回值为true或false

${status.last}   判断当前项是否为集合中的最后一项，返回值为true或false

begin、end、step分别表示：起始序号，结束序号，跳跃步伐。

如：<c:forEach begin='1' end='5' step='2' items='${list}' var='item'>

表示：操作list集合汇中1~5条数据，不是逐条循环，而是按每2个取值。即操作集合中的第1、3、5条数据。

5 . <c:set var="a" value="1"/>是什么意思这个是C标签的赋值  
不是从哪里来 而是你想要它显示到哪里

| **属性** | **描述** | **必需** | **默认** |
| --- | --- | --- | --- |
| value | 要保存的信息 | 否 | body |
| target | 要修改其属性的变量的名称 | 否 | None |
| property | 要修改的属性 | 否 | None |
| var | 用于存储信息的变量的名称 | 否 | None |
| scope | 变量存储信息范围 | 否 | Page |

例子：

<%@ page language="java" contentType="text/html; charset=UTF-8"

pageEncoding="UTF-8"%>

<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>c:set示例</title>

</head>

<body>

<c:set var="salary" scope="session" value="${5600\*2}" />

薪水值：

<c:out value="${salary}" />

</body>

</html>

结果：

薪水值：11200

# 6. jsp：bean标签中scope属性的四种属性值理解

这四个属性值所不同的就是usebean的生命周期。其中个人感觉而言page和request区别不大，下面来简要谈谈page、application、session的区别。

1、page的生命周期是创建对象开始，到本页执行结束。当下次在执行本页时（比如刷新操作）又重新创建，执行结束后有，这个对象所占的资源被释放。

2、session的生命周期是创建对象开始，到本次会话结束。什么是会话？可以说就是从第一次运行这个程序，到关闭服务器这段时间。打个形象的比方，当你打开浏览器，在地址栏输入：http:\\localhost:8080\......时，就好比要要会话的两个人见面了，不过在这儿，这两个人一个是tomcat服务器，一个是浏览器。因为运行JSP程序无非就是和tomcat服务器之间进行信息交流，所以第一次运行这个jsp：bean所在的文件，就相当于两个人进行了第一次交流。只有当这个浏览器关闭，相当于一方转身，这次会话才算结束，直到此时，对象资源才会被释放。

3、application的生命周期是服务器其停止间的时间。

理解了上面这些，就不那么理解下面这个现象：如下程序所示：

<%@ page language="java" contentType="text/html; charset=UTF-8"  
    pageEncoding="UTF-8"%>

<jsp:useBean id="myDate" scope="page" class="java.util.Date">  
  这是一个标签体   
</jsp:useBean>

当时page、request属性时，每次刷新都会有 “这是一个标签体”出现。

当时session属性时，执行第一次有 “这是一个标签体”，刷新之后就没有了。要想重新出现，要将浏览器重启一次。

当时session属性时，执行第一次有 “这是一个标签体”，刷新之后就没有了。要想重新出现，要将tomcat服务器重启一次。

7．现在可以娱乐一下哦

要验证一下<c:import>能否动态地引入不同的文件

**一、实验一**

前提：写了两个jsp文件：m.jsp  test1.jsp

目的：用变量"a"代替test1中的“1”

①m.jsp中

<@import prefix="c" uri="<http://java.sun.com/jstl.core>">

<c:set  var="a" value="1"/>

<c:import url="test${a}.jsp"/>

②test1.jsp中

<@import prefix="c" uri="<http://java.sun.com/jstl.core>">

<BR>TEST1<BR><c:out value="${a}"/>

结果：

TEST

1

那么，问题来了：如果给"a"设不同的值，能引入不同的文件吗？就有了

**二、实验二**

**①m.jsp中**

<@import prefix="c" uri="<http://java.sun.com/jstl.core>"> <c:set  var="a" value="1"/>

<c:import url="test${a}.jsp"/>

<c:set  var="a" value="2" scope="request"/>

<c:import url="test${a}.jsp"/>

**②test1.jsp中**

<@import prefix="c" uri="<http://java.sun.com/jstl.core>">

 <BR>TEST1<BR><c:out value="${a}"/>

**③test2.jsp中**

<@import prefix="c" uri="<http://java.sun.com/jstl.core>">

<BR>TEST2<BR><c:out value="${a}"/>

**结果如下:**

**TEST1**

**1**

**TEST1**

**1**

**三、实验三**

①怎么m.jsp中第二次设值要加一个“scope”属性呢？我试着把第一次设a的值也加上“scope="request"”，跑了跑。

结果：

TEST1

1

TEST2

2

②两行代码的scope都去掉，结果和都加上scope是一样的

**四、解释：**

默认下scope属性值是"page"，  <c:set  var="a" value="1"/>　 <c:set  var="a" value="2" scope="request"/>是申明了两个的不同作用域的变量，我们可以称为page.a ，request.a。

<c:import url="test${a}.jsp"/>引入时，肯定是先在当前页面找a的值了，找不到的话再去找request作用域的a，如果找到了，当然就不用去更大的作用域找了。

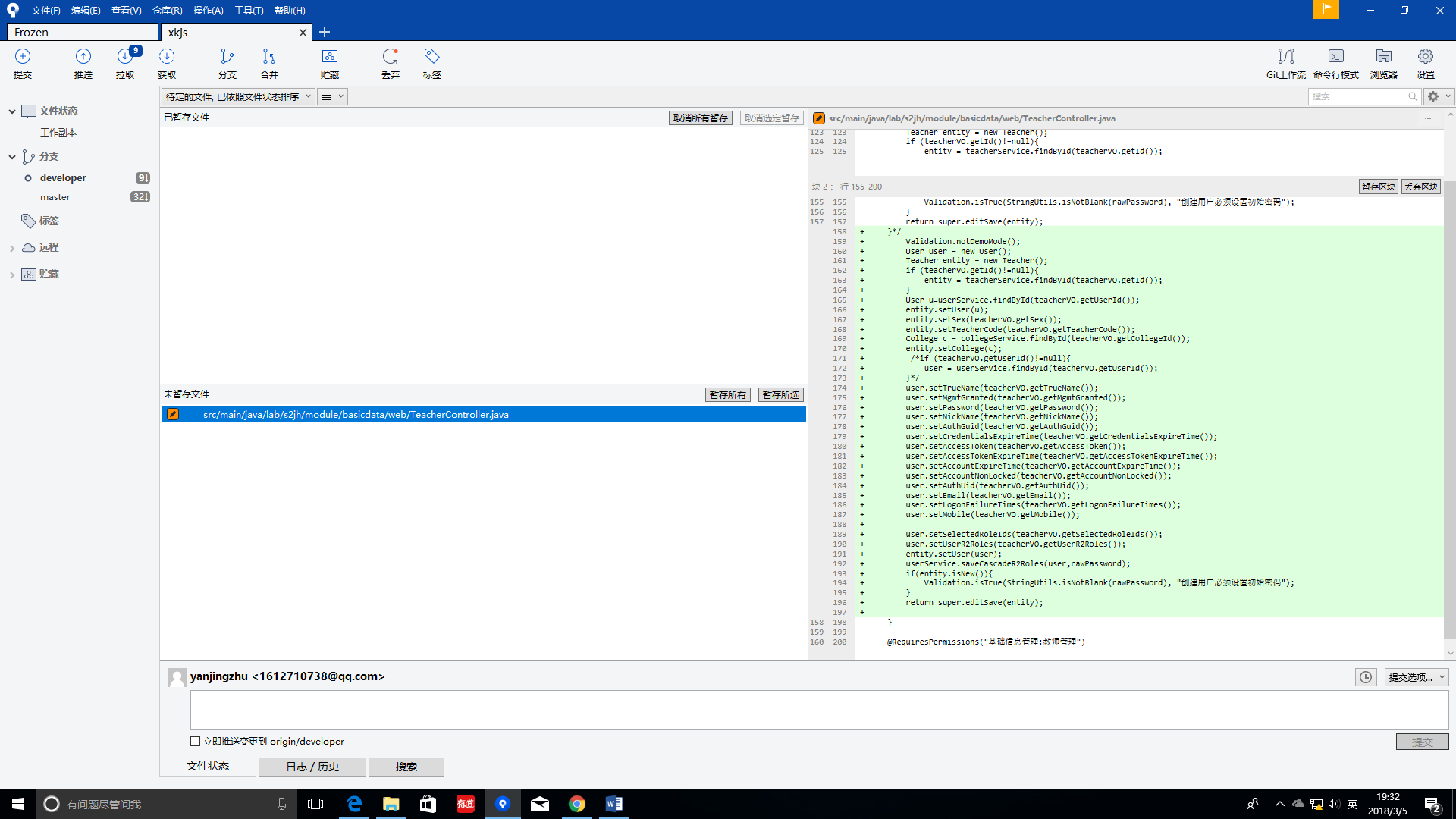
**五、总结：**

１.scope的作用域大小依次为：application>session>request>page(默认)

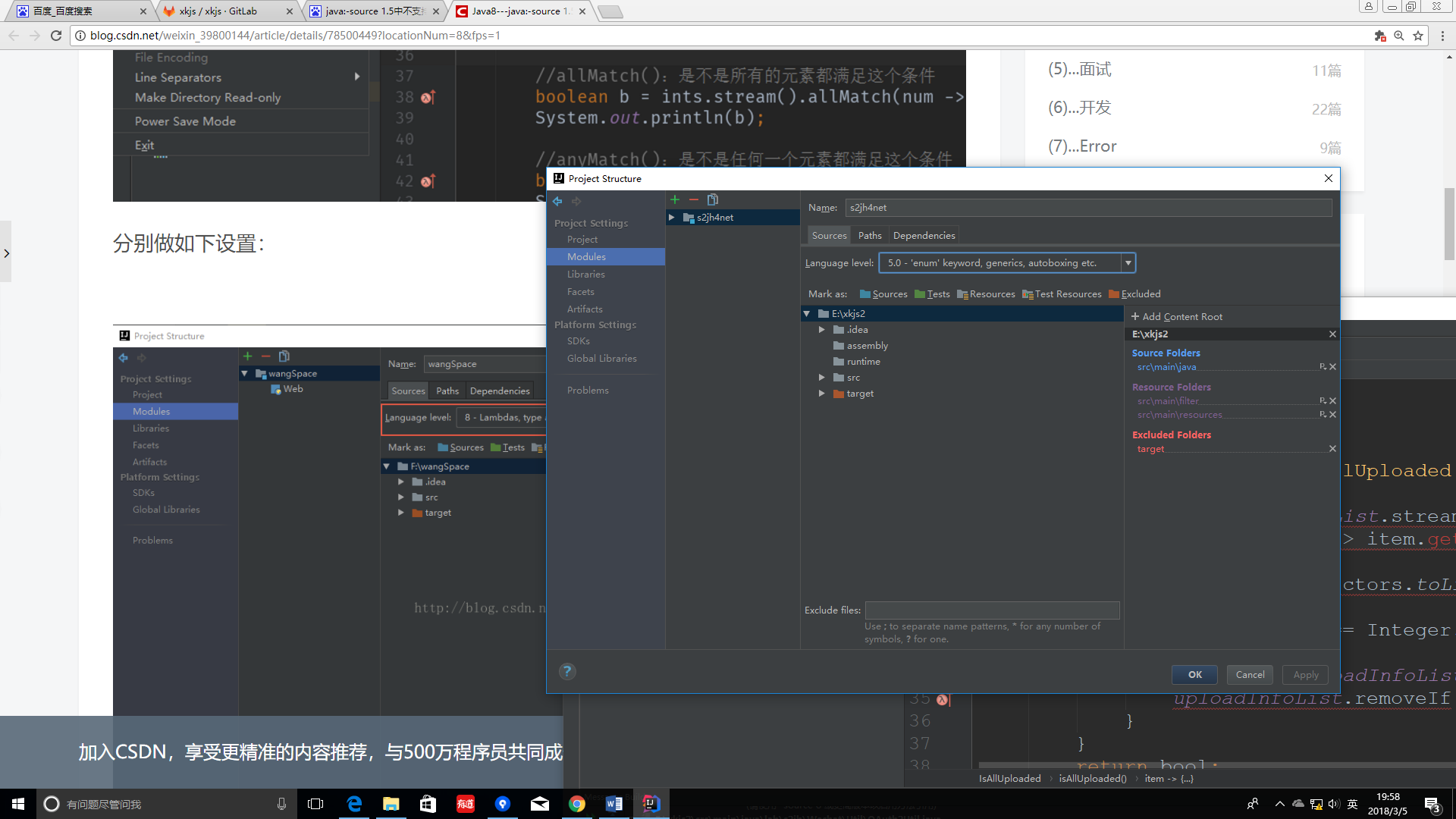
２.jsp处理变量的作用域先后依次为：page(默认)->request->session->application

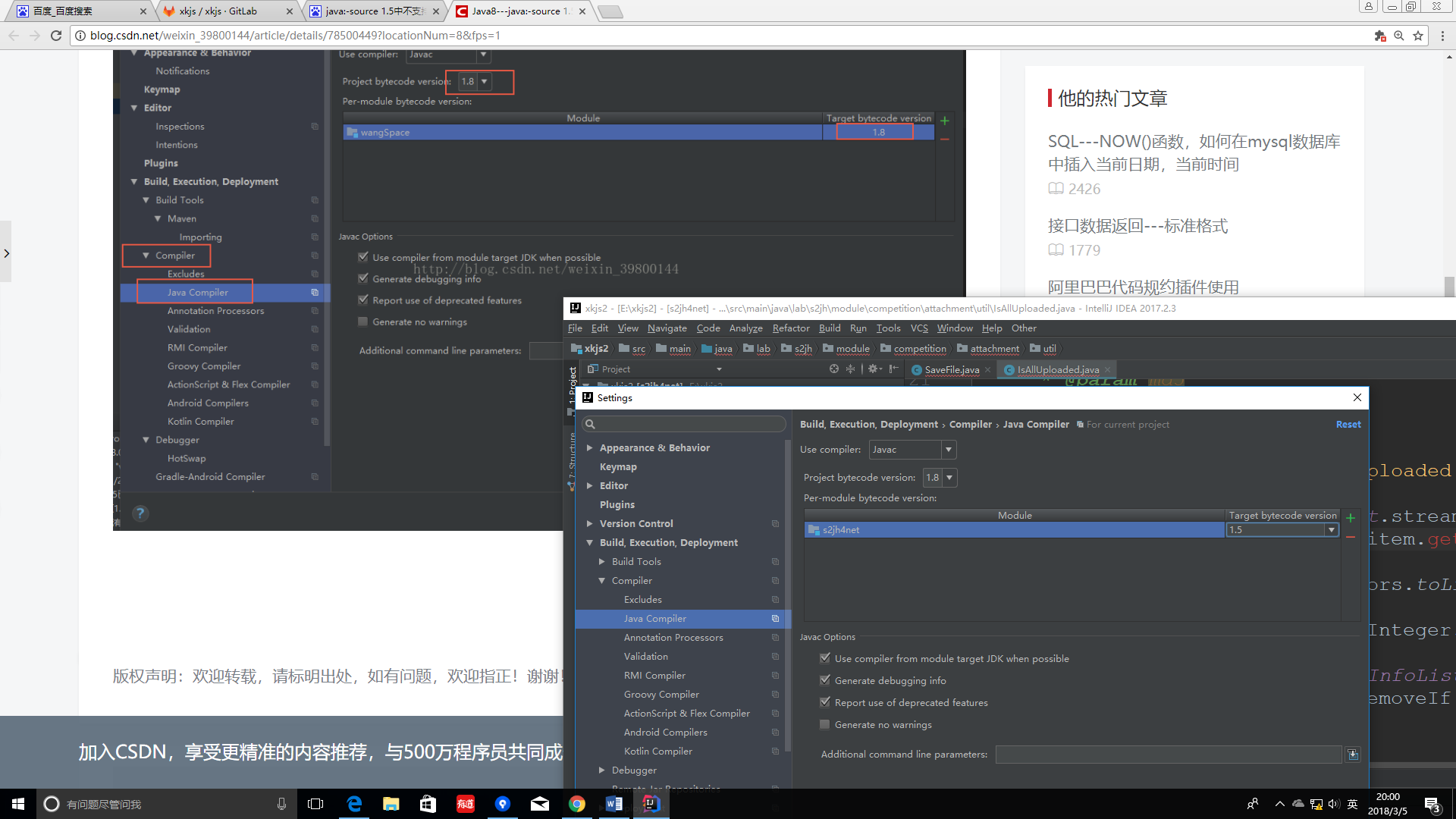
（2018/01/22）

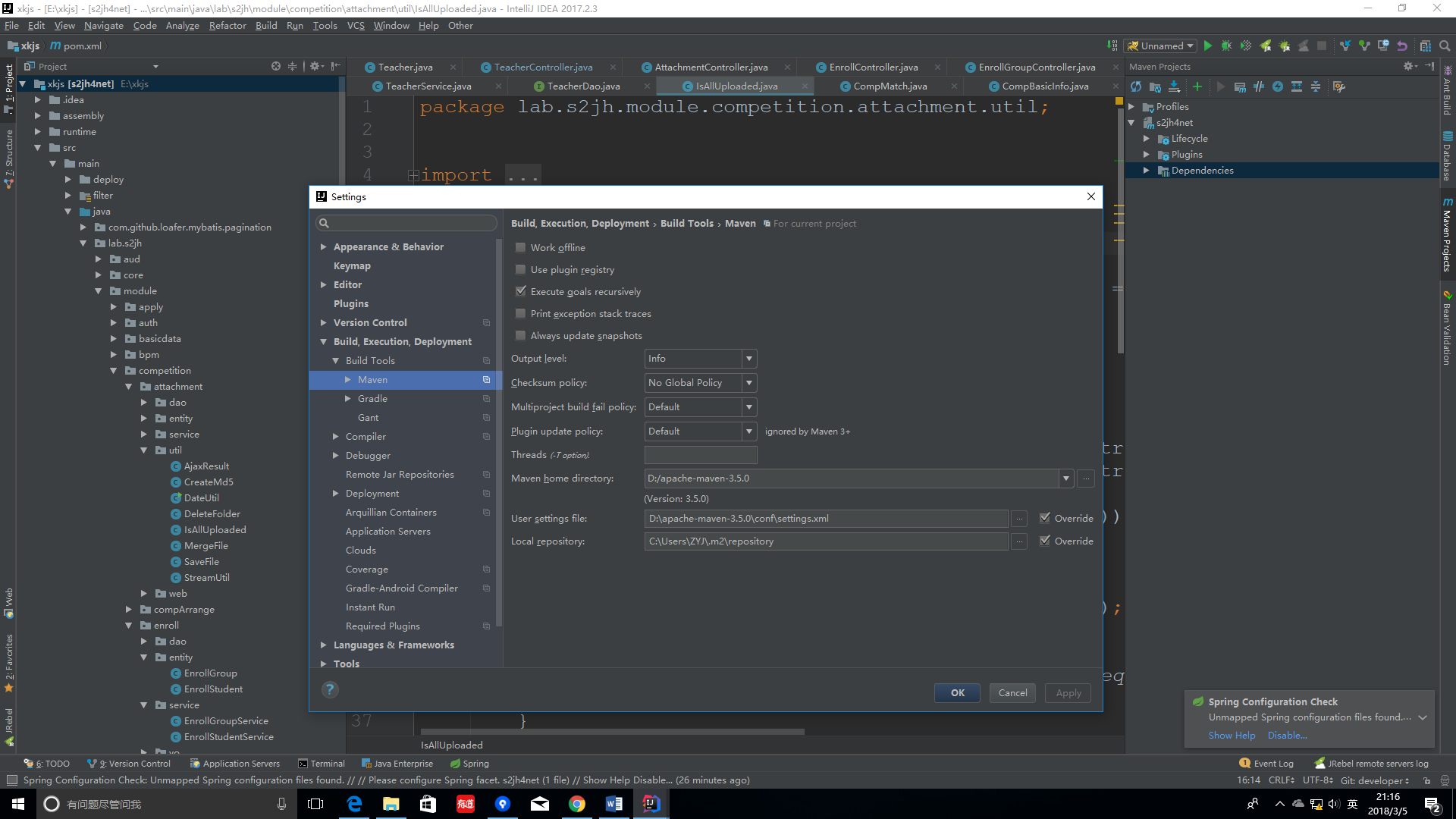
1. style="border-top: 1px solid black;"
2. Ctrl + PrScrn 使用这个[组合键](https://www.baidu.com/s?wd=%E7%BB%84%E5%90%88%E9%94%AE&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1d9mWTdrHf3PyDsmhRvryDz0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnWTLP1n4P1Dd)截屏，获得的是整个屏幕的图片
3. 之前改过的：



（2018/3/5）







<https://www.bilibili.com/>

<http://www.mybatis.org/mybatis-3/zh/index.html>

<https://www.bilibili.com/video/av14854688/>

（2018/3/9）

.1.

