Java web开发环境： Eclipse Neon+Tomcat8.5+MySQL5

2017.10.25

|  |  |
| --- | --- |
| **JavaBean属性命名问题** | |
| 报错内容 | javax.el.PropertyNotFoundException: Property 'isLead' not found on type java.lang.Boolean |
| 问题分析 | 命名javaBean 中 boolean 类型属性的时候千万小心啊！  private boolean isExist，这么写就直接悲剧了，因为自动生成的getter setter 是  　　public boolean isExist() { return isExist; } public void setExist(boolean isExist) { this.isExist = isExist; }  在EL表达式中取 xxx.isExist 是取不到值的！！！！  xxx.Exist 才可以取到  所以命名javaBean 中 boolean 类型属性 一定不要 is打头！！！ |
| 解决办法 | 一般JavaBean属性以小写字母开头，驼峰命名格式，相应的 getter/setter 方法是 get/set 接上首字母大写的属性名。例如：属性名为userName，其对应的getter/setter 方法是 getUserName/setUserName。  但是，还有一些特殊情况：  1、如果属性名的第二个字母大写，那么该属性名直接用作 getter/setter 方法中 get/set 的后部分，就是说大小写不变。例如属性名为uName，方法是getuName/setuName。  2、如果前两个字母是大写（一般的专有名词和缩略词都会大写），也是属性名直接用作 getter/setter 方法中 get/set 的后部分。例如属性名为URL，方法是getURL/setURL。  3、如果首字母大写，也是属性名直接用作 getter/setter 方法中 get/set 的后部分。例如属性名为Name，方法是getName/setName，这种是最糟糕的情况，会找不到属性出错，因为默认的属性名是name。  所以在JavaBean命名时应该注意符合以上命名规范。  需要注意的是有些开发工具的自动生成的getter/setter方法，并没有考虑到上面所说的特例情况，导致bug的产生，比如Eclipse3.4版本，Eclipse3.5较3.4智能些，避免了一些问题。以下是我在Eclipse3.5和Eclipse3.4分别做的实验：  在User的实体类中，设了两个变量，一个是userName，另一个是uName，然后用Eclipse的自动生成的get和set方法（source–>Generate getter andsetter），在Eclipse3.5中生成的是下图所示的正确的方法，正常的方法是将变量名的首字母变成大写，而如果变量名的第一个字母是小写，第二个字母是大写，则第一个字母为小写不变。 |
| 参考链接 | <http://blog.csdn.net/mccand1234/article/details/50804575>  <http://blog.csdn.net/liwen19840617/article/details/5060845>  <http://blog.csdn.net/yunye114105/article/details/7364264> |

|  |  |
| --- | --- |
| **Java.util.Date与Java.sql.Date类型转换问题** | |
| 报错内容 | java.util.Date cannot be cast to java.sql.Date |
| 问题分析 |  |
| 解决办法 | String employeeBirthDate = request.getParameter("employeeBirth");  SimpleDateFormat sdf = **new** SimpleDateFormat("yyyy-MM-dd");  Date employeeBirth = **null**;  **try** {  employeeBirth = sdf.parse(employeeBirthDate);  } **catch** (ParseException e) {  // **TODO** Auto-generated catch block  e.printStackTrace();  } |
| 参考链接 | <http://bbs.csdn.net/topics/390358278> |

|  |  |
| --- | --- |
| **Jsp中的中文传到后台是乱码** | |
| 报错内容 | 插入到数据库中的中文字段是乱码 |
| 问题分析 | 浏览器将数据(假设为“中国”）发送给服务器的时候，将数据变成0101的二进制数据（假设为98 99）时必然要查码表，浏览器以哪个码表打开网页，浏览器就以哪个码表提交数据。数据到达服务器后，数据（98 99）要封装到request中，在servlet中调用Request的getParameter方法返回的是字符串（“中国”），方法内部拿到数字后要转成字符，一定要查码表，由于request的设计者是外国人，所以默认查的是他们常用的ISO8859-1，这就是请求数据产生乱码的根源。 |
| 解决办法 | 1. 如果提交方式为post，想不乱码，只需要设置request对象的编码即可。   request.setCharacterEncoding("UTF-8");  注意：客户机数据是以哪种方式提交的，request就应该设成什么编码。  2.如果提交方式为get，设置request对象的编码是无效的，想不乱码，只能手工转换。  String data = "???????";//乱码字符串  byte source [] = data.getBytes("iso8859-1");//得到客户机提交的原始数据  data = new String (data.getBytes("iso8859-1"),"UTF-8");//解决乱码  //等同于  data = new String (source,"UTF-8"); |
| 参考链接 | http://www.jb51.net/article/56701.htm |

|  |  |
| --- | --- |
| **通过代理上外网，配置本地仓库与GitHub的连接方法。** | |
| 报错内容 | ssh: Could not resolve hostname github.com: Name or service not known  fatal: The remote end hung up unexpectedly |
| 问题分析 | 如果在公司内通过代理上网或者因为某些已知的非可抗力因素导致我们上网必须通过代理，用本地客户端访问github时，如果不做配置的话通过ssh来克隆repository的时候，用原生命令是不能直接访问的，需要配置下代理。 |
| 解决办法 | #回归家目录  thinker@thinker-young MINGW64 ~/.ssh  $ cd  # 查看家目录在windows中的位置  thinker@thinker-young MINGW64 ~  $ pwd  /c/Users/thinker  #切到ssh的配置文件下  thinker@thinker-young MINGW64 ~  $ cd .ssh  # 创建配置文件  thinker@thinker-young MINGW64 ~/.ssh  $ vi config  #写入下面的内容  thinker@thinker-young MINGW64 ~/.ssh  $ cat config  Host github.com  ProxyCommand ~/.ssh/ssh-https-tunnel %h %p  Port 443  Hostname ssh.github.com  #保存退出  thinker@thinker-young MINGW64 ~/.ssh      打开刚才下载ssh-https-tunnel这个文件，完成代理服务器ip和端口，及用户名和密码的配置： |
| 参考链接 | <http://blog.csdn.net/x6_9x/article/details/60873021>  <https://zwitterion.org/software/ssh-https-tunnel/ssh-https-tunnel> |