在Struts框架下使用时间类型 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/462/2021_2022__E5_9C_A8Struts_E6_c104_462420.htm 使用时间类型？这谁不会，不就是java.util下的几个类吗，在不加上java.sql和java.text下的几个类，这会有什么问题吗？Struts要是连时间都处理不了，那还能干嘛？ 在实际应用中，我就发现Struts确实连有些简单的时间都处理不了（不知是我使用的方法不对还是Struts确实没有考虑到）。顺便你也能了解Struts是怎么把form里的请求参数populate到ActionForm里面的。 今天下午同事告诉我把有java.util.Date类型属性的类存入数据库时出错，把这个属性删除就没有问题了。当时我就想到是RequestProcessor在processPopulate()时出错了，因此在它的这个方法设了断点并跟踪了进去。当然，它最先要调用ActionForm的reset()方法，然后调用实际处理populate（将请求参数传给ActionForm）的RequestUtils.populate（）方法。RequestUtils的这个静态方法最先是处理Multipart的（即文件上传等多部分）的方法，然后将所有的请求都放在叫properties的HashMap里并循环处理它： names = request.getParameterNames(). while (names.hasMoreElements()) { String name = (String) names.nextElement(). String stripped = name. if (prefix != null) { if (!stripped.startsWith(prefix)) { continue. } stripped = stripped.substring(prefix.length()). } if (suffix != null) { if (!stripped.endsWith(suffix)) { continue. } stripped = stripped.substring(0, stripped.length() - suffix.length()). } if (isMultipart) { properties.put(stripped,

multipartParameters.get(name)). } else { properties.put(stripped, request.getParameterValues(name)). } } 实际处理它们的是下面的：BeanUtils.populate(bean, properties). 其中bean就是接受数据的ActionForm，而properties里面则是所有的请求的键－值对（键和值都是字符串，http协议的特点）。 再看看BeanUtils的静态（类）方法populate是怎么处理的： // Loop through the property name/value pairs to be set Iterator names = properties.keySet().iterator(). while (names.hasNext()) { // Identify the property name and value(s) to be assigned String name = (String) names.next(). if (name == null) { continue. } Object value = properties.get(name). // Perform the assignment for this property setProperty(bean, name, value). } 它是循环所有的请求参数，把实际的工作又交给了setProperty方法。呵呵，弄了半天，这帮人原来都是代理。 这个方法还是代理吗？计算了一下它有180行的代码。这么长应该是个实干家了吧，错！千万不要被有些人的外表欺骗了！有些人一天上班16个小时，可够敬业的，可有8小时在打CS。这个类就是：一上来20多行都在一个if (log.isTraceEnabled()) {} 里面。 log在这说明一下。Struts中使用的是Jakarta Commons Logging的包，它使用的优先级是：Log4j（4念four好像比较有意义，大概是Logger For Java的意思，我听有的人年Log si J，感觉很别扭，呵呵），Java 1.4 Logging API，Simple Logging。功能是依次减弱。 建议在写Action 的execute()或被execute()调用的业务方法中使用Commons Logging 来代替System.out.println()－－当要你把成百上千的System.out.println()去掉的时候你就会觉得Commons Logging是个多好的东东了。它的用法是： import

org.apache.commons.logging.Log. import org.apache.commons.logging.LogFactory. private/protected static Log log = LogFactory.getLog(DispatchAction.class). 如果你用的是DispatchAction，那你就不要自己定义Log的实例了，因为它已经有一个protected的Log实例，直接使用即可。 使用方法是： if (log.isInfoEnabled()) { log.Info("some information."). } Logging把消息分为6种级别，debug,error,fatal,info,trace,warn。比如，你想记录一条消息，它只是为了给用户一个警告，则可以使用warn。为什么在每个log.Info()前做一次判断呢？难道如果log级别不允许Info，log.Info()仍然能Info吗？当然不是。它的作用是提高效率。 比如有个消息是计算前一万个自然数的和（这种消息可能少见）。用直接log.Info() int sum=0. for(int i=0.i sum =i. } log.Info("the sum of form 1 to 10000 is : "_sum). 如果log.Info是不允许的，那求10000个数的和就白求的。当然如果你的计算机很快或和高斯一样聪明，直接log.Info（）也每什么问题。 闲话少说，回到180多行的BeanUtils.setProperty（）方法。这个方法先是处理nested属性，也就是xxx.xxx的请求参数。我们只看看处理简单属性的必须过程。下面这端代码有点长，但它只做了一件事：将字符串的请求参数转成ActionForm的类型。比如：你在ActionForm里有个Integer userAge；然后HTTP请求参数里可能会有http://localhost:8080/xxx.do?userAge=21。传人的是字符串，目标是专程Integer。 首先它当然会根据userAge这个字符串查找相应的ActionForm，如果这个ActionForm有个属性也叫userAge，然后就会把这个userAge的类型存到type里，type的定义是：Class type = null. 得到type的代码很长，这是因为要它

考虑很多情况，例如DynaActionForm。 // Convert the specified value to the required type Object newValue = null. if (type.isArray() amp. (index if (value == null) { String values[] = new String[1]. values[0] = (String) value. newValue = ConvertUtils.convert((String[]) values, type). } else if (value instanceof String) { String values[] = new String[1]. values[0] = (String) value. newValue = ConvertUtils.convert((String[]) values, type). } else if (value instanceof String[]) { newValue = ConvertUtils.convert((String[]) value, type). } else { newValue = value. } } else if (type.isArray()) { // Indexed value into array if (value instanceof String) { newValue = ConvertUtils.convert((String) value, type.getComponentType()). } else if (value instanceof String[]) { newValue = ConvertUtils.convert(((String[]) value)[0], type.getComponentType()). } else { newValue = value. } } else { // Value into scalar if ((value instanceof String) || (value == null)) { newValue = ConvertUtils.convert((String) value, type). } else if (value instanceof String[]) { newValue = ConvertUtils.convert(((String[]) value)[0], type). } else if (ConvertUtils.lookup(value.getClass()) != null) { newValue = ConvertUtils.convert(value.toString(), type).// Here is my programs break point } else { newValue = value. } } 最后是：调用PropertyUtils的一些方法设置值。下面代码的第一种情况是有索引的，即你在请求参数里传了field[0]＝123之类的参数，第二种是Map类型的，传的是map(key)＝value之类的参数，最一般的就是调用第三个方法。 if (index >= 0) { PropertyUtils.setIndexedProperty(target, propName, index,

newValue). } else if (key != null) {
PropertyUtils.setMappedProperty(target, propName, 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com