[**关于json-lib There is a cycle in the hierarchy!问题的3种解决办法**](http://williambryantliu.blog.hexun.com/86791580_d.html) [原创 2013-7-11 15:22:15]

[我顶](http://cache-sidebar.blog.hexun.com/voteArticle.aspx?articleID=86791580&blogname=williambryantliu&blogid=2399691) 字号：大 **中** 小

问题：

net.sf.json.JSONException: There is a cycle in the hierarchy!

 at net.sf.json.util.CycleDetectionStrategy$StrictCycleDetectionStrategy.handleRepeatedReferenceAsObject(CycleDetectionStrategy.java:97)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

在查问一些资料后，总结出解决该问题的两种办法，现在与大家分享一下。

其一：根据原理来解决，如果需要解析的数据间存在级联关系，而互相嵌套引用，在hibernate中极容易嵌套而抛出net.sf.json.JSONException: There is a cycle in the hierarchy异常。

举个例子：现在有实验(Lib)和类别(Libtype)两张表，每个实验都对应着一个类别，那么，在类别的POJO中，就会如下代码:

 private Integer ltid; //类别ID

 private String ltype;  //类别名称

 private Set libs = new HashSet(0);  //对应的实验集

当我们写如下代码时，会报错：

 public ActionForward execute(ActionMapping mapping, ActionForm form,  HttpServletRequest request, HttpServletResponse response){

 LibtypeDAO libtypeDAO = new LibtypeDAO();

 List<Libtype> list = libtypeDAO.findAll();

 JSONArray jsonArray = JSONArray.fromObject(list);

 return null;

}

原因很简单，在Libtype中，有一个与List无关的属性值，即libs，我们只需要ltid和ltype，所以报错。

根据我在网上查找的资料，解决办法有如下3种：

1.设置JSON-LIB让其过滤掉引起循环的字段：

public ActionForward execute(ActionMapping mapping, ActionForm form, HttpServletRequest request, HttpServletResponse response){

  LibtypeDAO libtypeDAO = new LibtypeDAO();

  List<Libtype> list = libtypeDAO.findAll();

  JsonConfig jsonConfig = new JsonConfig();  //建立配置文件

  jsonConfig.setIgnoreDefaultExcludes(false);  //设置默认忽略

  jsonConfig.setExcludes(new String[]{"libs"});  //此处是亮点，只要将所需忽略字段加到数组中即可，在上述案例中，所要忽略的是“libs”，那么将其添到数组中即可，在实际测试中，我发现在所返回数组中，存在大量无用属性，如“multipartRequestHandler”，“servletWrapper”，那么也可以将这两个加到忽略数组中.

  JSONArray jsonArray = JSONArray.fromObject(list,jsonConfig);  //加载配置文件

  return null;

}

2.设置JSON-LIB的setCycleDetectionStrategy属性让其自己处理循环，省事但是数据过于复杂的话会引起数据溢出或者效率低下。

public ActionForward execute(ActionMapping mapping, ActionForm form, HttpServletRequest request, HttpServletResponse response){

LibtypeDAO libtypeDAO = new LibtypeDAO();

List<Libtype> list = libtypeDAO.findAll();

JsonConfig jsonConfig = new JsonConfig(); //建立配置文件

jsonConfig.setIgnoreDefaultExcludes(false); //设置默认忽略

jsonConfig.setCycleDetectionStrategy(CycleDetectionStrategy.LENIENT);   //此处是亮点，不过经过测试，第2种方法有些悲剧，虽然可以使用，但其结果貌似循环数次，至于为啥，还请高人指点。

JSONArray jsonArray = JSONArray.fromObject(list,jsonConfig); //加载配置文件

return null;

}

3.最为原始的办法，自己写个JavaBean，用forEach循环，添加到List中，这个方法我看网上有人成功，我没试，但大概过程可以写出来，其结果正确性有待检验。

JavaBean:

public LibtypeForm{

int ltid;

string ltname;

}

public ActionForward execute(ActionMapping mapping, ActionForm form, HttpServletRequest request, HttpServletResponse response){

LibtypeDAO libtypeDAO = new LibtypeDAO();

List<Libtype> list = libtypeDAO.findAll();

List<LibtypeForm> formList = new ArrayList();

for(Libtype libtype : list){

LibtypeForm form = new LibtypeForm();

form.setLtid(libtype .getLtid);

form.setLtname(libtype.getLtname);

formList.add(form);

}

JSONArray jsonArray = JSONArray.fromObject(formList);

return null;

}