**package** com.sinosoft.microservice.channel.common.utils;  
  
**import** java.lang.reflect.Constructor;  
**import** java.lang.reflect.Field;  
**import** java.lang.reflect.Method;  
**import** java.util.ArrayList;  
**import** java.util.HashMap;  
**import** java.util.List;  
**import** java.util.Map;  
  
*/\*\*  
 \** ***@author*** *sh on 2018/6/10  
 \** ***@version*** *ideaIU-2017.2.5.win  
 \*/***public class** EntityUtils {  
  
 */\*\*  
 \* 将数组数据转换为实体类   
 \* 此处数组元素的顺序必须与实体类构造函数中的属性顺序一致   
 \*  
 \** ***@param list*** *数组对象集合   
 \** ***@param clazz*** *实体类   
 \** ***@param <T>*** *实体类   
 \** ***@param model*** *实例化的实体类   
 \** ***@return*** *实体类集合  
 \*/* **public static** <T> List<T> castEntity(List<Object[]> list, Class<T> clazz, Object model) {  
 List<T> returnList = **new** ArrayList<T>();  
 **if** (list.isEmpty()) {  
 **return** returnList;  
 }  
 *//获取每个数组集合的元素个数* Object[] co = list.get(0);  
  
 *//获取当前实体类的属性名、属性值、属性类别* List<Map> attributeInfoList = *getFiledsInfo*(model);  
 *//创建属性类别数组* Class[] c2 = **new** Class[attributeInfoList.size()];  
 *//如果数组集合元素个数与实体类属性个数不一致则发生错误* **if** (attributeInfoList.size() != co.**length**) {  
 **return** returnList;  
 }  
 *//确定构造方法* **for** (**int** i = 0; i < attributeInfoList.size(); i++) {  
 c2[i] = (Class) attributeInfoList.get(i).get(**"type"**);  
 }  
 **try** {  
 **for** (Object[] o : list) {  
 Constructor<T> constructor = clazz.getConstructor(c2);  
 returnList.add(constructor.newInstance(o));  
 }  
 } **catch** (Exception ex) {  
   
 **return** returnList;  
 }  
 **return** returnList;  
 }  
  
 */\*\*  
 \* 根据属性名获取属性值   
 \*  
 \** ***@param fieldName*** *属性名   
 \** ***@param modle*** *实体类   
 \** ***@return*** *属性值  
 \*/* **private static** Object getFieldValueByName(String fieldName, Object modle) {  
 **try** {  
 String firstLetter = fieldName.substring(0, 1).toUpperCase();  
 String getter = **"get"** + firstLetter + fieldName.substring(1);  
 Method method = modle.getClass().getMethod(getter, **new** Class[]{});  
 Object value = method.invoke(modle, **new** Object[]{});  
 **return** value;  
 } **catch** (Exception e) {  
 **return null**;  
 }  
 }  
  
 */\*\*  
 \* 获取属性类型(type)，属性名(name)，属性值(value)的map组成的list   
 \*  
 \** ***@param model*** *实体类   
 \** ***@return*** *list集合  
 \*/* **private static** List<Map> getFiledsInfo(Object model) {  
 Field[] fields = model.getClass().getDeclaredFields();  
 List<Map> list = **new** ArrayList(fields.**length**);  
 Map infoMap = **null**;  
 **for** (**int** i = 0; i < fields.**length**; i++) {  
 infoMap = **new** HashMap(3);  
 infoMap.put(**"type"**, fields[i].getType());  
 infoMap.put(**"name"**, fields[i].getName());  
 infoMap.put(**"value"**, *getFieldValueByName*(fields[i].getName(), model));  
 list.add(infoMap);  
 }  
 **return** list;  
 }  
}