```
package com.prototype.deepclone;

import java.io.ByteArrayInputStream;
import java.io.ByteArrayOutputStream;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.io.Serializable;

public class DeepProtoType implements Serializable, Cloneable{

  public String name; //String 属性
  public DeepCloneableTarget deepCloneableTarget;// 引用类型
  public DeepProtoType() {
    super();
  }


  //深拷贝 - 方式 1 使用clone 方法
  @Override
  protected Object clone() throws CloneNotSupportedException {

    Object deep = null;
    //这里完成对基本数据类型(属性)和String的克隆
    deep = super.clone();
    //对引用类型的属性，进行单独处理
    DeepProtoType deepProtoType = (DeepProtoType)deep;
    deepProtoType.deepCloneableTarget  =
(DeepCloneableTarget)deepCloneableTarget.clone();

    // TODO Auto-generated method stub
    return deepProtoType;
  }

  //深拷贝 - 方式2 通过对象的序列化实现 (推荐)

  public Object deepClone() {

    //创建流对象
    ByteArrayOutputStream bos = null;
    ObjectOutputStream oos = null;
    ByteArrayInputStream bis = null;
    ObjectInputStream ois = null;

    try {

      //序列化
```

```
            bos = new ByteArrayOutputStream();
            oos = new ObjectOutputStream(bos);
            oos.writeObject(this); //当前这个对象以对象流的方式输出

            //反序列化
            bis = new ByteArrayInputStream(bos.toByteArray());
            ois = new ObjectInputStream(bis);
            DeepProtoType copyObj = (DeepProtoType)ois.readObject();

            return copyObj;

        } catch (Exception e) {
            // TODO: handle exception
            e.printStackTrace();
            return null;
        } finally {
            //关闭流
            try {
                bos.close();
                oos.close();
                bis.close();
                ois.close();
            } catch (Exception e2) {
                // TODO: handle exception
                System.out.println(e2.getMessage());
            }
        }

    }

}
```