**长沙学院计算机工程与应用数学学院**

**实验报告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 课程名称 | 面向对象程序设计 | 班级 | 20软件 | 实验日期 | 4月4日 |
| 姓名 | 李佳骏 | 学号 | B20190103224 | 指导教师 | 黄志坚 |
| 成绩 |  | | | | |

一. 实验名称

JAVA序列化及反序列化

二. 实验目的及要求

1）熟练使用对象的序列化编写相应应用程序

三. 实验环境

Eclipse IDE

四. 实验内容

（1）学生成绩及成绩保存到本地磁盘“save.txt”中。

（2）学生的身份证号禁止序列化。

（3）运行时如果不存在“save.txt”，则重新录入学生的信息与成绩。

（4）如果存在“save.txt”，则显示学生信息与成绩。

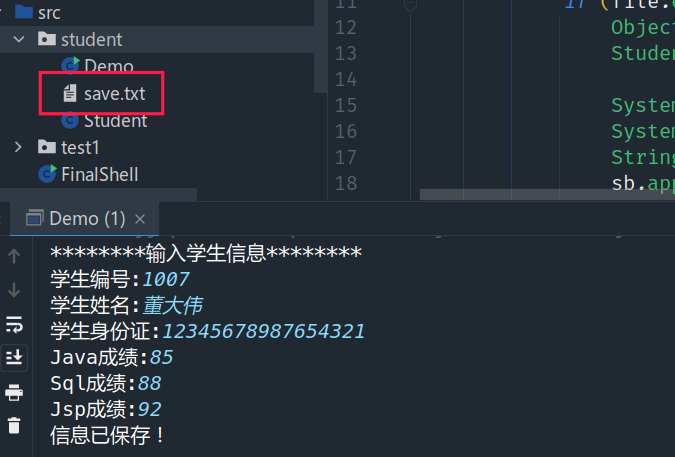
1. 实验过程及实验结果
2. 创建学生类

*package* student;  
  
*import* java*.*io*.*Serializable;  
  
*public class* Student *implements* Serializable {  
 *private* String id;  
 *private* String name;  
 *private transient* String card;  
 *private* Double java;  
 *private* Double sql;  
 *private* Double jsp;  
  
 *public* Student() {  
 }  
  
 *public* Student(String id, String name, String card, Double java, Double sql, Double jsp) {  
 *this.*id = id;  
 *this.*name = name;  
 *this.*card = card;  
 *this.*java = java;  
 *this.*sql = sql;  
 *this.*jsp = jsp;  
 }  
  
 *public* String getId() {  
 *return* id;  
 }  
  
 *public void* setId(String id) {  
 *this.*id = id;  
 }  
  
 *public* String getName() {  
 *return* name;  
 }  
  
 *public void* setName(String name) {  
 *this.*name = name;  
 }  
  
 *public* String getCard() {  
 *return* card;  
 }  
  
 *public void* setCard(String card) {  
 *this.*card = card;  
 }  
  
 *public* Double getJava() {  
 *return* java;  
 }  
  
 *public void* setJava(Double java) {  
 *this.*java = java;  
 }  
  
 *public* Double getSql() {  
 *return* sql;  
 }  
  
 *public void* setSql(Double sql) {  
 *this.*sql = sql;  
 }  
  
 *public* Double getJsp() {  
 *return* jsp;  
 }  
  
 *public void* setJsp(Double jsp) {  
 *this.*jsp = jsp;  
 }  
  
 @Override  
 *public* String toString() {  
 *return* "Student{" +  
 "id='" + id + '\'' +  
 ", name='" + name + '\'' +  
 ", card='" + card + '\'' +  
 ", java=" + java +  
 ", sql=" + sql +  
 ", jsp=" + jsp +  
 '}';  
 }  
  
 *public* Double sum() {  
 *return* java + sql + jsp;  
 }  
}

1. 创建测试类

*package* student;  
  
*import* java*.*io*.*\*;  
*import* java*.*util*.*Scanner;  
  
*public class* Demo {  
 *public static void* main(String[] args) {  
 *try* {  
 File file = *new* File("./序列化/src/student/save.txt");  
 *if* (file*.*exists()) {  
 ObjectInputStream ois = *new* ObjectInputStream(*new* FileInputStream(file));  
 Student student = (Student) ois*.*readObject();  
  
 System*.*out*.*println("×××学生基本信息×××");  
 System*.*out*.*println("编号 姓名 身份证");  
 StringBuilder sb = *new* StringBuilder();  
 sb*.*append(student*.*getId())*.*append(" ")*.*append(student*.*getName())*.*append(" ")*.*append(student*.*getCard());  
 System*.*out*.*println(sb);  
 System*.*out*.*println();  
 System*.*out*.*println("×××学生成绩×××");  
 System*.*out*.*println("科目 成绩");  
 System*.*out*.*println("Java " + student*.*getJava());  
 System*.*out*.*println("Sql " + student*.*getSql());  
 System*.*out*.*println("Jsp " + student*.*getJsp());  
 System*.*out*.*println("总成绩 " + student*.*sum());  
 } *else* {  
 Student st;  
 Scanner sc = *new* Scanner(System*.*in);  
 String id, name, card;  
 *double* java, sql, jsp;  
 System*.*out*.*println("\*\*\*\*\*\*\*\*输入学生信息\*\*\*\*\*\*\*\*");  
 System*.*out*.*print("学生编号:");  
 id = sc*.*next();  
 System*.*out*.*print("学生姓名:");  
 name = sc*.*next();  
 System*.*out*.*print("学生身份证:");  
 card = sc*.*next();  
 System*.*out*.*print("Java成绩:");  
 java = sc*.*nextDouble();  
 System*.*out*.*print("Sql成绩:");  
 sql = sc*.*nextDouble();  
 System*.*out*.*print("Jsp成绩:");  
 jsp = sc*.*nextDouble();  
  
 st = *new* Student(id, name, card, java, sql, jsp);  
 FileOutputStream fos = *new* FileOutputStream(file);  
 ObjectOutputStream oos = *new* ObjectOutputStream(fos);  
 oos*.*writeObject(st);  
 fos*.*close();  
 oos*.*close();  
 System*.*out*.*println("信息已保存！");  
 }  
 } *catch* (Exception e) {  
 System*.*out*.*println("保存失败！请重新运行程序！");  
 e*.*printStackTrace();  
 }  
 }  
}

1. 测试截图





六. 实验小结

* 1. 当父类继承Serializable接口时，所有子类都可以被序列化。
* 2.子类实现了Serializable接口，父类没有，父类中的属性不能序列化（不报错，数据会丢失），但是在子类中属性仍能正确序列化。
* 3.如果序列化的属性是对象，则这个对象也必须实现Serializable接口，否则会报错。
* 4.在反序列化时，如果对象有属性的修改或删减，则修改的部分属性会丢失，但不会报错。
* 5.在反序列化时，如果serialVersionUID被修改，则反序列化会失败。

七. 教师评阅意见