**天津工业大学**

**计算机科学与软件学院**

**实习报告**

课 程 名 称: **《面向对象应用实习》**

实 习 题 目:MVC架构的信息管理系统设计与实现

专 业 班 级: 计算机1604

学 生 姓 名: 鲁海

学　　 号: 1611640422

实 习 成 绩:

指导教师签名： 年 月 日

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 实习时间 | | 2017.12.25-2018.1.3 | 实习地点 | C205 | 实习学时 | 30 |
| 实习  目的 | 以 Java 语言为例逐步设计开发一个基于 MVC 架构、具备 GUI 界面的信息管理系统。课程采用循序渐 进方式，从设计简单的 Java 类开始，逐步增加功能、通过使用设计模式对代码 进行重构，分阶段完成 3 个里程碑版本：基于内存中集合对象的命令行信息管理 系统、基于文件的命令行信息管理系统和 MVC 架构的 GUI 信息管理系统。通过这 些针对性练习使学生充分理解面向对象程序设计的思想及其应用方式。 | | | | | |
| 实习  内容 | 利用Java语言，采用MVC架构设计实现一个人员信息管理系统。程序具备GUI界面，并要求人员信息以文本形式存储在指定文件中。 | | | | | |
| 实  习  要  求 | 阶段1：实现基于Java集合对象的命令行信息管理系统  阶段2：信息存储在文本文件中，程序能够自动存取文件中的信息。  阶段3：采用GUI界面；信息存储在文本文件中；采用MVC架构进行重构设计 | | | | | |
| 实习  主要  设施  设备 | 计算机 | | | | | |
| 实  习  成  果 | 采用MVC架构设计实现了一个具备GUI界面的人员信息管理系统，并且  可以将人员信息以文本形式存储在指定文件中。 | | | | | |

|  |
| --- |
| 实习报告（提纲和格式参考如下~） |
| **1 引言**  1.1系统概述（二级标题，四号黑体，左对齐）  利用Java语言，采用MVC架构设计实现一个人员信息管理系统。程序具备GUI界面，并要求人员信息以文本形式存储在指定文件中。  1.2 需求分析  对学生成员进行管理。  1.3 运行环境  开发环境：Windows 10  JDK:1.8.0\_121  开发工具:Eclipse  Eclipse版本： Oxygen Release (4.7.0)  **2 概要设计**  2.1 设计思路  首先将学生信息管理系统的功能全部实现。利用MVC将model层定义好，定义controller调用model方法。设计两个GUI主界面，添加各种按钮和文本区，将要实现的功能选项添加其中，然后对这些选项实施监听，实现其功能。  2.2 模块功能介绍（针对MVC三层分别介绍）  视图层：用户看到并与之交互的界面。  模型层：一个模型能为多个视图提供数据  控制层：接受用户的输入并调用模型和视图去完成用户的需求，控制器本身不输出任何东西和做任何处理。它只是接收请求并决定调用哪个模型构件去处理请求，然后再确定用哪个视图来显示返回的数据。  2.3 模块结构图  控制层：    模块层：    视图层：    2.4 程序流程图    **3　详细设计**（小三黑体，居中）  InfoSys应用包括三个核心接口。  InfoSysView：视图层的接口，负责生成与用户交互的图形界面。  InfoSysModel: 控制器层的接口，负责调用模型和视图。  InfoSysController\_stu: 模型层的接口，负责处理业务逻辑，访问文件。  如例程1 所示是 InfoSysView接口的源程序。它包括以下 3 个方法。  ◆add UserGestureListener(InfoSysController\_stu ctrl)方法：在视图中注册处理各种用户动  作（比如用户按下【查询客户】按钮）的控制器，参数 ctrl 指定控制器。  ◆showDisplay(Object display)方法：在图形界面上显示数据，参数 display 指定待显示  的数据。  ◆handleStudentChange ()方法：当模型层修改了文件中某个客户的信息时，同步刷新  视图的图形界面。  例程1：  package InfoSys.view;  import java.rmi.Remote;  import java.rmi.RemoteException;  import InfoSys.InfoSysException;  import InfoSys.controller.InfoSysController\_stu;  import InfoSys.vo.Student;  public interface InfoSysView extends Remote  {  /\*\*注册处理用户动作的监听器\*/  public void addUserGestureListener(InfoSysController\_stu ctrl)throws InfoSysException,RemoteException;  /\*\*在图形界面上显示数据，参数display指定显示的数据\*/  public void showDisplay(Object display)throws InfoSysException,RemoteException ;  public void handleStudentChange(Student cust)throws InfoSysException,RemoteException;  }  例程2：InfoSysController\_stu.java  package InfoSys.controller;  import java.util.List;  import InfoSys.vo.Student;  public interface InfoSysController\_stu {  /\*\* 处理添加学生的动作 \*/  public void handleAddGesture(Student stu);  /\*\* 处理刷新学生的动作 \*/  public void handleShowGesture(Student stu) throws Exception;  /\*\* 处理根据ID查询学生的动作 \*/  public void handleFindByIDGesture(String id) throws Exception;  /\*\* 处理根据名字查询学生的动作 \*/  public void handleFindByNameGesture(String name) throws Exception;  /\*\* 处理删除学生的动作 \*/  public void handleDelGesture(Student stu) throws Exception;  /\*\* 处理修改学生的动作 \*/  public void handleModGesture(Student stu) throws Exception;  /\*\* 处理读取文件的动作 \*/  public Object handleReadGesture() throws Exception;  /\*\* 处理保存文件的动作 \*/  public void handleSaveGesture(List<Student> it) throws Exception;  /\*\* 处理获取全部学生的动作 \*/  public void handleGetAllStudent();    }  如例程3所示是 InfoSysModel接口的源程序。InfoSysModel接口中声明了操纵文件的一系列方法，这些方法用于添加、更新、删除和查询文件中的学生信息等。  例程3：InfoSysModel.java  package InfoSys.model;  import java.rmi.RemoteException;  import java.util.List;  import java.util.Set;  import InfoSys.InfoSysException;  import InfoSys.view.InfoSysView;  import InfoSys.vo.\*;  public interface InfoSysModel {  public void add(Student stu) throws InfoSysException,Exception;  public void show(Student stu) throws InfoSysException,Exception;  public Student findByID(String id) throws InfoSysException,Exception;    public Student findByName(String name) throws InfoSysException,Exception;  public void del(Student stu) throws InfoSysException,Exception;  public void mod(Student stu) throws InfoSysException,Exception;    public Object read() throws InfoSysException,Exception;    public void save(List<Student> it) throws InfoSysException,Exception;    public Set<Student> getAllStudent()throws InfoSysException,Exception;  /\*\*注册视图，以便当模型修改了数据库中的人员信息时可以回调视图的刷新界面方法\*/  public void addChangeListener(InfoSysView iv)throws RemoteException;  public void fireModelChangeEvent(Student stu);  }  Store Gui 类负责生成图形界面。如例程 4 所示是 Store Gui 类的源程  序。  例程4：Gui\_stu.java  package InfoSys.view;  import javax.swing.\*;  import javax.swing.table.DefaultTableModel;  import InfoSys.vo.Student;  import java.awt.\*;  import java.awt.event.ActionListener;  import java.util.Iterator;  import java.util.Set;  public class Gui\_stu extends JFrame {  //界面的主要窗体组件  protected JFrame frame;  protected Container contentPane;  protected CardLayout card = new CardLayout();  protected JPanel cardPan = new JPanel();    //包含各种按钮的选择面板上的组件  protected JPanel selPan = new JPanel();  protected JButton stuBt = new JButton("学生详细信息");  protected JButton allStuBt = new JButton("所有学生清单");    //显示单个学生的面板上的组件  protected JPanel stuPan=new JPanel();  protected JLabel idLb=new JLabel("ID");  protected JLabel nameLb=new JLabel("学生姓名");  protected JLabel ageLb=new JLabel("学生年龄");  protected JLabel scoreLb=new JLabel("学生成绩");    public static JTextField idTf=new JTextField(25);  public static JTextField nameTf=new JTextField(25);  public static JTextField ageTf=new JTextField(25);  public static JTextField scoreTf=new JTextField(25);    protected JButton addBt=new JButton("添加学生");  protected JButton delBt=new JButton("删除学生");  protected JButton findByIDBt=new JButton("按ID查找学生");  protected JButton findByNameBt=new JButton("按姓名关键字查找学生");  protected JButton modBt=new JButton("修改学生信息");  protected JButton showBt=new JButton("更新信息");    //列举所有学生的面板上的组件  protected JPanel allStuPan=new JPanel();  protected JLabel allStuLb=new JLabel("所有学生清单 ",SwingConstants.CENTER);  protected JTextArea allStuTa=new JTextArea();  protected JScrollPane allStuSp=new JScrollPane(allStuTa);  //table头  String[] tableHeaders={"ID","姓名","年龄","成绩"};  JTable table;  JScrollPane tablePane;  DefaultTableModel tableModel;  /\*\* 构造方法 \*/  public Gui\_stu(){  buildDisplay();  }  //日志面板上的组件  protected JPanel logPan=new JPanel();  protected JLabel logLb=new JLabel("操作日志 ",SwingConstants.CENTER);  protected static JTextArea logTa=new JTextArea(9,50);  protected JScrollPane logSp=new JScrollPane(logTa);    /\*\* 显示单个学生面板 StuPan \*/  public void refreshStuPan(Student stu){  showCard("student");  int index = table.getSelectedRow(); //获取选中行  stu.putID(table.getValueAt(index,0).toString());  stu.putName(table.getValueAt(index,1).toString());  stu.putAge(Integer.parseInt((String)table.getValueAt(index,2)));  stu.putScore(Float.parseFloat((String)table.getValueAt(index,3)));  if(stu==null || stu.getID().equals("-1")){  idTf.setText(null);  nameTf.setText(null);  ageTf.setText(null);  scoreTf.setText(null);  return;  }  idTf.setText(new Long(stu.getID()).toString());  nameTf.setText(stu.getName().trim());  ageTf.setText(String.valueOf(stu.getAge()));  scoreTf.setText(String.valueOf(stu.getScore()));  }    /\*\* 显示所有客户面板 allStuPan \*/  public void refreshAllStuPan(Set stus){  showCard("allstudents");    String newData[][];  newData=new String[stus.size()][4];  Iterator it=stus.iterator();  int i=0;  while(it.hasNext()){  Student stu=(Student) it.next();  newData[i][0]=new Integer(stu.getID()).toString();  newData[i][1]=stu.getName();  newData[i][2]=String.valueOf(stu.getAge());  newData[i][3]=String.valueOf(stu.getScore());  i++;  }  tableModel.setDataVector(newData,tableHeaders);  }    /\*\* 在日志面板 log Pan 中添加日志信息 \*/  public static void updateLog(String msg){  logTa.append(msg+"\n");  }    /\*\* 获得学生面板 stuPan 上用户输入的 ID \*/  public String getStuIdOnCustPan(){  try{  return idTf.getText().trim();  }catch(Exception e){  updateLog(e.getMessage());  return null;  }  }      /\*\* 显示单个客户面板 stuPan 或者所有客户面板 allStuPan \*/  void showCard(String cardStr){    card.show(cardPan,cardStr);  allStuPan.setVisible(true);  }        /\*\* 创建图形界面 \*/  private void buildDisplay(){  //frame=new JFrame("学生信息管理系统");  setTitle("学生信息管理系统");  buildSelectionPanel();  buildStuPanel();  buildAllStuPanel();  buildLogPanel();  /\*\* carPan 采用 CardLayout 布局管理器，包括 stuPan 和  allStuPan 两张卡片 \*/  cardPan.setLayout(card);  cardPan.add(stuPan,"student");  cardPan.add(allStuPan,"allstudents");  //向主窗体中加入各种面板    this.setLayout(new BorderLayout());  add(cardPan,BorderLayout.CENTER);  add(selPan,BorderLayout.NORTH);  add(logPan,BorderLayout.SOUTH);  this.pack();  this.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);  this.setVisible(true);    }  //获得单个人员面板上用户输入的人员信息  public Student getStudentOnCustPanel()  {  try  {  String custid=idTf.getText().trim();  String cid = custid.equals("")?"0":custid;  return new Student(cid,  nameTf.getText(),  Integer.parseInt(ageTf.getText().trim()),  Float.parseFloat(scoreTf.getText()));  }catch(Exception e)  {  updateLog(e.getMessage());  return null;  }  }  /\*\* 创建选择面板 selPan \*/  private void buildSelectionPanel(){  selPan.add(stuBt);  selPan.add(allStuBt);  }    /\*\* 为选择面板 selPan 中的两个按钮注册监听器 \*/  public void addSelectionPanelListeners(ActionListener a[]){  int len=a.length;  if(len!=2){ return;}  stuBt.addActionListener(a[0]);  allStuBt.addActionListener(a[1]);  }  public static void main(String[]args)  {  new Gui\_stu().setVisible(true);;    }  /\*\* 创建单个客户stuPan 面板 \*/  private void buildStuPanel(){  stuPan.setLayout(new GridLayout(7,2,2,2));  stuPan.add(idLb);  stuPan.add(idTf);  stuPan.add(nameLb);  stuPan.add(nameTf);  stuPan.add(ageLb);  stuPan.add(ageTf);  stuPan.add(scoreLb);  stuPan.add(scoreTf);  stuPan.add(addBt);  stuPan.add(delBt);  stuPan.add(findByIDBt);  stuPan.add(findByNameBt);  stuPan.add(modBt);  stuPan.add(showBt);  }    /\*\* 为单个客户面板 stuPan 中的6 个按钮注册监听器 \*/  public void addStuPanelListeners(ActionListener a[]){  addBt.addActionListener(a[0]);  delBt.addActionListener(a[1]);  findByIDBt.addActionListener(a[2]);  findByNameBt.addActionListener(a[3]);  modBt.addActionListener(a[4]);  showBt.addActionListener(a[5]);  }    /\*\* 创建所有学生 allStuPan 面板 \*/  private void buildAllStuPanel(){  allStuPan.setLayout(new BorderLayout());  allStuPan.add(allStuLb,BorderLayout.NORTH);  allStuTa.setText("all student display");  tableModel=new DefaultTableModel(tableHeaders,10);  table=new JTable(tableModel);  tablePane=new JScrollPane(table);  allStuPan.add(tablePane,BorderLayout.CENTER);  Dimension dim=new Dimension(500,150);  table.setPreferredScrollableViewportSize(dim); //设置此表视口的首选大小。  }    /\*\* 创建日志面板\*/  private void buildLogPanel(){  logPan.setLayout(new BorderLayout());  logSp.setHorizontalScrollBarPolicy(JScrollPane.HORIZONTAL\_SCROLLBAR\_ALWAYS);  logSp.setVerticalScrollBarPolicy(JScrollPane.VERTICAL\_SCROLLBAR\_ALWAYS);  logPan.add(logLb,BorderLayout.NORTH);  this.logPan.add(logSp,BorderLayout.CENTER); }      }  **4 调试分析**（小三黑体，居中）  问题：第一次添加学生信息时，如果文件中已存在相同ID的学生，仍然会添加成功。  解决办法：程序启动时就读取文本中的信息并储存在Arraylist中。  **5 测试**（小三黑体，居中）  启动图：    添加学生信息：        删除学生：      按ID查找学生：    按名字模糊查找：    修改学生信息：  修改前：      修改后：    **心得与体会**（小三黑体，居中）  通过这次实训，我深刻体会到了开发程序的不易，尤其是修复BUG。  同时，我也体会到了MVC架构的优点，即用一种业务逻辑、数据、界面显示分离的方法组织代码，将业务逻辑聚集到一个部件里面，在改进和个性化定制界面及用户交互的同时，不需要重新编写业务逻辑。在今后的编程中，我会多多运用MVC架构，争取熟练掌握。  **参考文献** （小三黑体，居中）  [1]百度：https://www.baidu.com/  [2]CSDN：http://blog.csdn.net/  [3]菜鸟教程: http://www.runoob.com/  **附 录**（小三黑体，居中） |