**实 验 报 告**

|  |  |
| --- | --- |
| **课程名称：** | **Java语言程序设计** |
| **学生姓名：** |  |
| **学生学号：** |  |
| **学生专业：** | **软件工程** |
| **开课学期：** | **2024-2025学年第一学期** |

**软件学院**

**2023年11月**

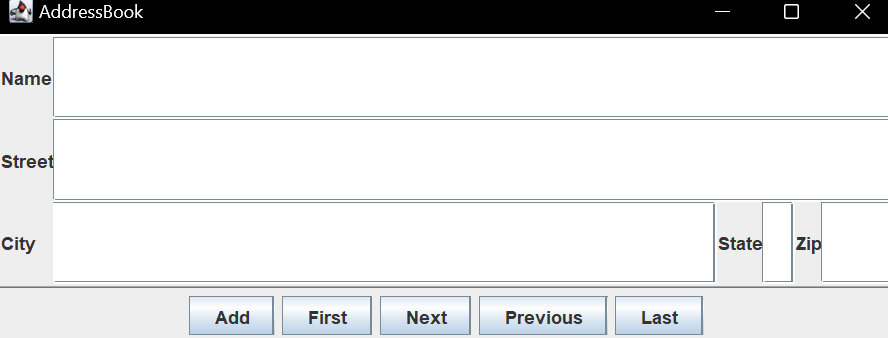
**实验四 GUI程序设计与I/O文件处理**

**一、实验目的**

1. 理解Java系统图形用户界面的工作原理和界面设计步骤；
2. 掌握图形界面容器（Frame、Panel）的使用；
3. 掌握主要布局管理器（FlowLayout、GridLayout、BorderLayout）的使用
4. 掌握图形用户界面的常用组件（JLabel、JTextField、JButton、JList）的使用方法；
5. 理解图形用户界面的事件响应机制，掌握ActionEvent事件编程方法；
6. 理解java IO的基本原理，掌握RandomAccessFile类使用方法。

**二、实验内容及结果**

**1.前置任务**

创建一个java工程，导入AddressBook.java 和FixedLengthStringIO.java文件，编译运行，观察源代码与运行结果

**2.step8**

**Step8.1：**新建banking.DAO 包，在该包中导入FixedLengthStringIO.java、并且新建CustomerDAO.java。CustomerDAO类封装了Customer文件访问的所有方法；

CustomerDAO代码如下：

1. **package** banking.DAO;
3. **import** banking.domain.Customer;
4. **import** banking.domain.Bank;
6. **import** java.io.\*;
7. **import** java.util.ArrayList;
9. **public** **class** CustomerDAO {
10. **private** **static** **final** String FILE\_NAME = "customers.dat";
12. **public** **void** saveCustomers(ArrayList<Customer> customers) {
13. **try** (RandomAccessFile raf = **new** RandomAccessFile(FILE\_NAME, "rw")) {
14. raf.setLength(0); // Clear the file before writing
15. **for** (Customer customer : customers) {
16. FixedLengthStringIO.writeFixedLengthString(customer.getFirstName(), 32, raf);
17. FixedLengthStringIO.writeFixedLengthString(customer.getLastName(), 32, raf);
18. }
19. } **catch** (IOException e) {
20. e.printStackTrace();
21. }
22. }
24. **public** ArrayList<Customer> loadCustomers() {
25. ArrayList<Customer> customers = **new** ArrayList<>();
26. **try** (RandomAccessFile raf = **new** RandomAccessFile(FILE\_NAME, "r")) {
27. **while** (raf.getFilePointer() < raf.length()) {
28. String firstName = FixedLengthStringIO.readFixedLengthString(32, raf).trim();
29. String lastName = FixedLengthStringIO.readFixedLengthString(32, raf).trim();
30. customers.add(**new** Customer(firstName, lastName));
31. }
32. } **catch** (IOException e) {
33. e.printStackTrace();
34. }
35. **return** customers;
36. }
37. }

**Step8.2：**修改Bank类，调用CustomerDAO类的方法，使其支持从文件读取客户或保存客户至文件

Bank代码如下：

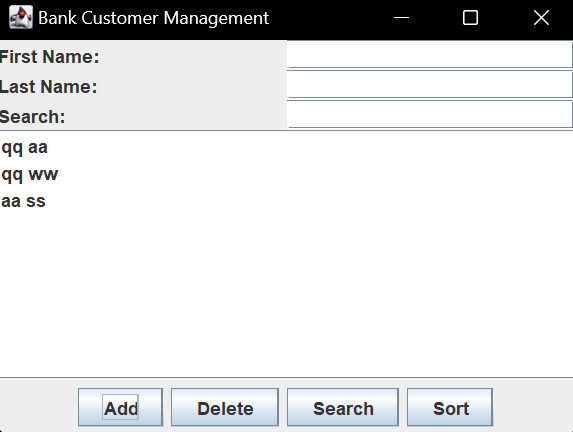
1. **package** banking.domain;
3. **import** java.util.ArrayList;
4. **import** java.util.Collections;
5. **import** java.util.List;
7. **public** **class** Bank {
8. **private** **static** Bank instance = **new** Bank();
9. **private** List<Customer> customers = **new** ArrayList<>();
11. **private** Bank() {}
13. **public** **static** Bank getBank() {
14. **return** instance;
15. }
17. **public** **int** getNumOfCustomers() {
18. **return** customers.size();
19. }
21. **public** Customer getCustomer(**int** index) {
22. **return** customers.get(index);
23. }
25. **public** **void** addCustomer(String firstName, String lastName) {
26. customers.add(**new** Customer(firstName, lastName));
27. }
29. **public** **void** sortCustomers() {
30. Collections.sort(customers, (c1, c2) -> c1.getLastName().compareTo(c2.getLastName()));
31. }
33. // 添加这个方法
34. **public** List<Customer> getCustomers() {
35. **return** customers;
36. }
37. }

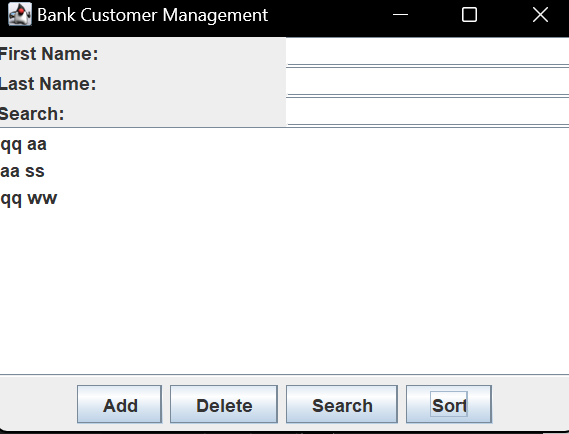
**Step8.3：**新建banking.UI 包，新建MainWindow.java, 调用Bank类的方法，处理界面组件的显示及相应按钮事件

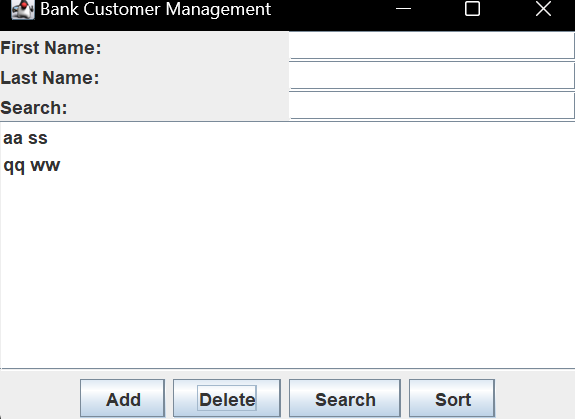
MainWindow.java代码如下：

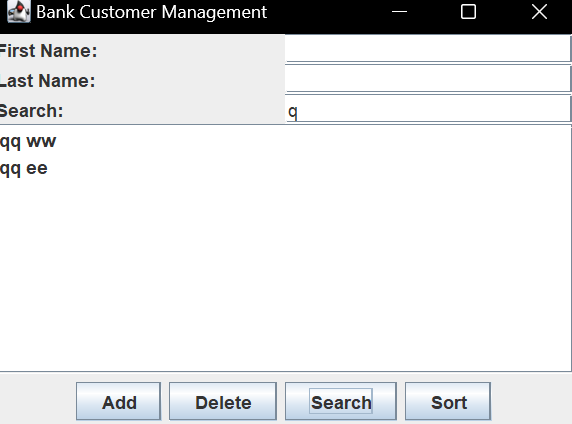
1. **package** banking.UI;
3. **import** banking.domain.Bank;
4. **import** banking.domain.Customer;
5. **import** banking.DAO.CustomerDAO;
7. **import** javax.swing.\*;
8. **import** javax.swing.border.BevelBorder;
9. **import** java.awt.\*;
10. **import** java.awt.event.ActionEvent;
11. **import** java.awt.event.ActionListener;
13. **public** **class** MainWindow **extends** JFrame {
14. **private** JList<String> customerList;
15. **private** DefaultListModel<String> listModel;
16. **private** JTextField firstNameField;
17. **private** JTextField lastNameField;
18. **private** JTextField searchField;
20. **public** MainWindow() {
21. setTitle("Bank Customer Management");
22. setSize(400, 300);
23. setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);
24. setLocationRelativeTo(**null**);
26. Bank bank = Bank.getBank();
27. listModel = **new** DefaultListModel<>();
28. **for** (**int** i = 0; i < bank.getNumOfCustomers(); i++) {
29. Customer customer = bank.getCustomer(i);
30. listModel.addElement(customer.getFirstName() + " " + customer.getLastName());
31. }
32. customerList = **new** JList<>(listModel);
33. JScrollPane scrollPane = **new** JScrollPane(customerList);
35. firstNameField = **new** JTextField(15);
36. lastNameField = **new** JTextField(15);
37. searchField = **new** JTextField(15);
39. JButton addButton = **new** JButton("Add");
40. JButton deleteButton = **new** JButton("Delete");
41. JButton searchButton = **new** JButton("Search");
42. JButton sortButton = **new** JButton("Sort");
44. JPanel inputPanel = **new** JPanel();
45. inputPanel.setLayout(**new** GridLayout(3, 2));
46. inputPanel.add(**new** JLabel("First Name:"));
47. inputPanel.add(firstNameField);
48. inputPanel.add(**new** JLabel("Last Name:"));
49. inputPanel.add(lastNameField);
50. inputPanel.add(**new** JLabel("Search:"));
51. inputPanel.add(searchField);
53. JPanel buttonPanel = **new** JPanel();
54. buttonPanel.add(addButton);
55. buttonPanel.add(deleteButton);
56. buttonPanel.add(searchButton);
57. buttonPanel.add(sortButton);
59. setLayout(**new** BorderLayout());
60. add(scrollPane, BorderLayout.CENTER);
61. add(inputPanel, BorderLayout.NORTH);
62. add(buttonPanel, BorderLayout.SOUTH);
64. addButton.addActionListener(**new** ActionListener() {
65. @Override
66. **public** **void** actionPerformed(ActionEvent e) {
67. String firstName = firstNameField.getText();
68. String lastName = lastNameField.getText();
69. **if** (!firstName.isEmpty() && !lastName.isEmpty()) {
70. Bank.getBank().addCustomer(firstName, lastName);
71. listModel.addElement(firstName + " " + lastName);
72. firstNameField.setText("");
73. lastNameField.setText("");
74. }
75. }
76. });
78. deleteButton.addActionListener(**new** ActionListener() {
79. @Override
80. **public** **void** actionPerformed(ActionEvent e) {
81. **int** selectedIndex = customerList.getSelectedIndex();
82. **if** (selectedIndex != -1) {
83. Bank.getBank().getCustomers().remove(selectedIndex);
84. listModel.remove(selectedIndex);
85. }
86. }
87. });
89. searchButton.addActionListener(**new** ActionListener() {
90. @Override
91. **public** **void** actionPerformed(ActionEvent e) {
92. String searchQuery = searchField.getText();
93. listModel.clear();
94. **for** (**int** i = 0; i < Bank.getBank().getNumOfCustomers(); i++) {
95. Customer customer = Bank.getBank().getCustomer(i);
96. **if** (customer.getFirstName().contains(searchQuery) || customer.getLastName().contains(searchQuery)) {
97. listModel.addElement(customer.getFirstName() + " " + customer.getLastName());
98. }
99. }
100. }
101. });
103. sortButton.addActionListener(**new** ActionListener() {
104. @Override
105. **public** **void** actionPerformed(ActionEvent e) {
106. Bank.getBank().sortCustomers();
107. listModel.clear();
108. **for** (**int** i = 0; i < Bank.getBank().getNumOfCustomers(); i++) {
109. Customer customer = Bank.getBank().getCustomer(i);
110. listModel.addElement(customer.getFirstName() + " " + customer.getLastName());
111. }
112. }
113. });
114. }
116. **public** **static** **void** main(String[] args) {
117. SwingUtilities.invokeLater(() -> {
118. MainWindow window = **new** MainWindow();
119. window.setVisible(**true**);
120. });
121. }
122. }

**运行结果：**

**添加信息**

**排序**

**选中后删除**

**根据关键词查找**

**三．实验问题回答**