**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**ГОСУДАРСТВЕННОЕ УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**«ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ БЕЛОРУССКОГО ГОСУДАРСТВЕННОГО УНИВЕРСИТЕТА»**

КАФЕДРА ИНФОРМАЦИОННЫХ СИСТЕМ УПРАВЛЕНИЯ

**ОТЧЕТ**

ПО УЧЕБНОЙ ПРАКТИКЕ

(ВАРИАНТ 6, ЛАБОРАТОРНАЯ РАБОТА 10)

Петрова Андрея Александрович

студента 2 курса, группа 14

специальность «Прикладная информатика»

**Руководитель**:

старший преподаватель Орешко И.Г.

Минск, 2021

## **ЗАДАНИЕ №10.**

**Цель работы:** Создаём собственный редактор для каждого свойства компонента. Каждый редактор ограничивает возможные значения свойства, предоставляя выбор из списка трёх – пяти допустимых значений (т. е. определяем методы getTags())

Регистрируем редакторы в классе BeanInfo компонента.

Дополнительно:

Попытайтесь создать настройщик компонента, который позволит изменять списки допустимых значений для свойств вашего компонента.

**Ход работы:**

Компонент Java Beans

**Манифест:**

Name: TextWithButtonPanelBean.class

Java-Bean: true

Код основного компонента:

public class TextWithButtonPanel extends Panel{  
 protected String messageText; // The message to display  
 protected Alignment alignment; // The alignment of the message  
 protected String simpleButtonLabel; // Text for the simpleButton button  
 protected String[] listText;  
  
 protected SingleLineLabel message;  
 protected Button simpleButton;  
 protected javax.swing.JList<String> list;  
public TextWithButtonPanel() {  
 this("Your Message Here");  
 }  
  
 public TextWithButtonPanel(String messageText) {  
 this(messageText, new String[] {"item1", "item2", "item3", "item4"});  
 }  
  
 public TextWithButtonPanel(String messageText, String[] listText) {  
 this(messageText, listText, "OK");  
 }  
  
 */\*\* A constructor for programmers using this class "by hand" \*/* public TextWithButtonPanel(String messageText, String[] listText, String simpleButtonLabel)  
 {  
 setLayout(new BorderLayout(15, 15));  
 message = new SingleLineLabel(messageText);  
 add(message, BorderLayout.NORTH);  
 Panel buttonbox = new Panel();  
 buttonbox.setLayout(new FlowLayout(FlowLayout.CENTER, 25, 15));  
 add(buttonbox, BorderLayout.SOUTH);  
 simpleButton = new Button(); // Create buttons  
 buttonbox.add(simpleButton);  
 simpleButton.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e) {  
 fireEvent(new AcceptEvent(TextWithButtonPanel.this));  
 }  
 });  
  
 list = new javax.swing.JList<>();  
 add(list, BorderLayout.CENTER);  
 setMessageText(messageText);  
 setSimpleButtonLabel(simpleButtonLabel);  
 setListText(listText);  
 }  
 public String getMessageText() { return messageText; }  
 public String getSimpleButtonLabel() { return simpleButtonLabel; }  
 public Alignment getAlignment() { return alignment; }  
 public String[] getListText() { return listText; }  
  
 public void setMessageText(String messageText) {  
 this.messageText = messageText;  
 message.setLabel(messageText);  
 validate();  
 }  
  
 public void setAlignment(Alignment alignment) {  
 this.alignment = alignment;  
 message.setAlignment(alignment);  
 }  
  
 public void setListText(String[] listText) {  
 this.listText = listText;  
 list.setListData(listText);  
 validate();  
 }  
  
 public void setSimpleButtonLabel(String l) {  
 simpleButtonLabel = l;  
 simpleButton.setLabel(l);  
 simpleButton.setVisible((l != null) && (l.length() > 0));  
 validate();  
 }  
  
 public void setFont(Font f) {  
 super.setFont(f); // Invoke the superclass method  
 message.setFont(f);  
 simpleButton.setFont(f);  
 list.setFont(f);  
 validate();  
 }protected Vector<AcceptListener> listeners = new Vector<>();public void addAcceptListener(AcceptListener l) {  
 listeners.addElement(l);  
 }public void removeAcceptListener(AcceptListener l) {  
 listeners.removeElement(l);  
 }public void fireEvent(AcceptEvent e) {  
 // Make a copy of the list and fire the events using that copy.  
 // This means that listeners can be added or removed from the original  
 // list in response to this event. We ought to be able to just use an  
 // enumeration for the vector, but that doesn't actually copy the list.  
 Vector list = (Vector) listeners.clone();  
 for(int i = 0; i < list.size(); i++) {  
 AcceptListener listener = (AcceptListener)list.elementAt(i);  
 AcceptEvent: listener.simpleButton(e); break;  
 }  
 }public static void main(String[] args) throws IOException {  
 // Create an instance of InfoPanel, with title and message specified:  
 TextWithButtonPanel p = new TextWithButtonPanel("Do you really want to quit?");  
  
 p.addAcceptListener(new AcceptListener() {  
 public void simpleButton(AcceptEvent e) {  
 System.out.printf("Selected row index: %d", p.list.getLeadSelectionIndex());  
 JOptionPane.showMessageDialog(p, "Selected items index = " +   
 p.list.getLeadSelectionIndex());  
 p.setMessageText("Selected row index: " + p.list.getLeadSelectionIndex());  
 System.exit(0);  
 }  
 });  
  
 Frame f = new Frame();  
 f.add(p);  
 f.pack();  
 f.setVisible(true);  
 }  
}

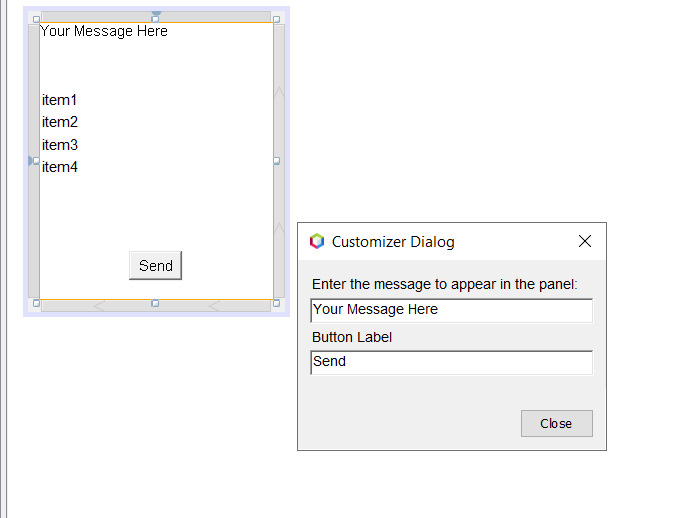
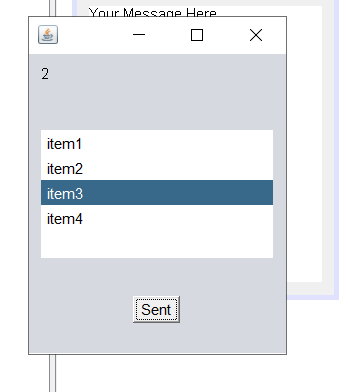


Рисунок 1. Результат выполнения задания №10