package SwingSamples;

import java.awt.EventQueue;

import javax.swing.JFrame;

import javax.swing.JPanel;

import javax.swing.border.EmptyBorder;

import javax.swing.\*;

import javax.swing.table.DefaultTableModel;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.ArrayList;

public class JCLass{

private JFrame frame;

private JLabel titleLabel;

private JTable contactTable;

private DefaultTableModel tableModel;

private JButton addButton;

private JButton editButton;

private JButton deleteButton;

private JComboBox<String> cboCategory;

private JCheckBox emailCheckBox;

private JCheckBox addressCheckBox;

private JList<String> categoryList;

private DefaultListModel<String> categoryListModel;

private JTextField nameField;

private JTextField phoneNumberField;

private ArrayList<Contact> contactList;

public JCLass{ () {

initialize();

}

private void initialize() {

// Initialize the main frame

frame = new JFrame("Contact Class");

frame.getContentPane().setBackground(new Color(128, 128, 64));

frame.setBounds(100, 100, 582, 463);

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.getContentPane().setLayout(null);

// Title label

titleLabel = new JLabel("Contact Table");

titleLabel.setFont(new Font("Tahoma", Font.BOLD, 16));

titleLabel.setBounds(10, 11, 150, 25);

frame.getContentPane().add(titleLabel);

// Contact table

tableModel = new DefaultTableModel(new Object[]{"Name", "Category", "Phone Numbers"}, 0);

contactTable = new JTable(tableModel);

JScrollPane tableScrollPane = new JScrollPane(contactTable);

tableScrollPane.setBounds(120, 46, 438, 201);

frame.getContentPane().add(tableScrollPane);

// Add button

addButton = new JButton("Add");

addButton.setFont(new Font("Tahoma", Font.BOLD, 10));

addButton.setBackground(new Color(0, 128, 255));

addButton.setBounds(20, 119, 90, 25);

addButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

addContact();

}

});

frame.getContentPane().add(addButton);

// Edit button

editButton = new JButton("Edit");

editButton.setFont(new Font("Tahoma", Font.BOLD, 10));

editButton.setBackground(new Color(0, 128, 255));

editButton.setBounds(20, 190, 90, 25);

editButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

editContact();

}

});

frame.getContentPane().add(editButton);

// Delete button

deleteButton = new JButton("Delete");

deleteButton.setFont(new Font("Tahoma", Font.BOLD, 10));

deleteButton.setBackground(new Color(0, 128, 255));

deleteButton.setBounds(20, 155, 90, 25);

deleteButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

deleteContact();

}

});

frame.getContentPane().add(deleteButton);

// Category combo box

cboCategory = new JComboBox<String>();

cboCategory.setBackground(new Color(128, 128, 128));

cboCategory.setBounds(118, 297, 150, 25);

frame.getContentPane().add(cboCategory);

// Email checkbox

emailCheckBox = new JCheckBox("Has Email");

emailCheckBox.setBackground(new Color(192, 192, 192));

emailCheckBox.setBounds(120, 341, 100, 25);

frame.getContentPane().add(emailCheckBox);

// Address checkbox

addressCheckBox = new JCheckBox("Has Address");

addressCheckBox.setBackground(new Color(192, 192, 192));

addressCheckBox.setBounds(243, 341, 120, 25);

frame.getContentPane().add(addressCheckBox);

// Category list

categoryListModel = new DefaultListModel<String>();

JScrollPane listScrollPane = new JScrollPane();

listScrollPane.setBounds(20, 46, 90, 63);

frame.getContentPane().add(listScrollPane);

categoryList = new JList<String>(categoryListModel);

listScrollPane.setViewportView(categoryList);

categoryList.setBackground(new Color(128, 255, 255));

// Name field

nameField = new JTextField();

nameField.setBackground(new Color(192, 192, 192));

nameField.setBounds(118, 262, 150, 25);

frame.getContentPane().add(nameField);

// Phone number field

phoneNumberField = new JTextField();

phoneNumberField.setBackground(new Color(192, 192, 192));

phoneNumberField.setBounds(278, 262,170, 25);

frame.getContentPane().add(phoneNumberField);

// Initialize the contact list

contactList = new ArrayList<Contact>();

// Add example categories

addCategory("Student");

addCategory("Teacher");

addCategory("Parent");

// Display the frame

frame.setVisible(true);

}

private void addCategory(String category) {

cboCategory.addItem(category);

categoryListModel.addElement(category);

}

private void addContact() {

String name = nameField.getText();

String category = (String) cboCategory.getSelectedItem();

String phoneNumber = phoneNumberField.getText();

boolean hasEmail = emailCheckBox.isSelected();

boolean hasAddress = addressCheckBox.isSelected();

// Validate input

if (name.isEmpty() || phoneNumber.isEmpty()) {

JOptionPane.showMessageDialog(frame, "Please enter name and phone number.", "Error",

JOptionPane.ERROR\_MESSAGE);

return;

}

// Create new contact

Contact contact = new Contact(name, category, phoneNumber, hasEmail, hasAddress);

contactList.add(contact);

// Update the table

tableModel.addRow(new Object[]{contact.getName(), contact.getCategory(),

contact.getPhoneNumber()});

// Clear input fields

nameField.setText("");

phoneNumberField.setText("");

emailCheckBox.setSelected(false);

addressCheckBox.setSelected(false);

}

private void editContact() {

int selectedRow = contactTable.getSelectedRow();

// Check if a contact is selected

if (selectedRow == -1) {

JOptionPane.showMessageDialog(frame, "Please select a contact to edit.", "Error",

JOptionPane.ERROR\_MESSAGE);

return;

}

// Get the selected contact

Contact contact = contactList.get(selectedRow);

// Update the input fields with the selected contact's information

nameField.setText(contact.getName());

cboCategory.setSelectedItem(contact.getCategory());

phoneNumberField.setText(contact.getPhoneNumber());

emailCheckBox.setSelected(contact.hasEmail());

addressCheckBox.setSelected(contact.hasAddress());

// Remove the selected contact from the list

contactList.remove(selectedRow);

// Remove the selected row from the table

tableModel.removeRow(selectedRow);

}

private void deleteContact() {

int selectedRow = contactTable.getSelectedRow();

// Check if a contact is selected

if (selectedRow == -1) {

JOptionPane.showMessageDialog(frame, "Please select a contact to delete.", "Error",

JOptionPane.ERROR\_MESSAGE);

return;

}

// Remove the selected contact from the list

contactList.remove(selectedRow);

// Remove the selected row from the table

tableModel.removeRow(selectedRow);

}

public static void main(String[] args) {

SwingUtilities.invokeLater(new Runnable() {

public void run() {

new Main();

}

});

}

private class Contact {

private String name;

private String category;

private String phoneNumber;

private boolean hasEmail;

private boolean hasAddress;

public Contact(String name, String category, String phoneNumber, boolean hasEmail, boolean

hasAddress) {

this.name = name;

this.category = category;

this.phoneNumber = phoneNumber;

this.hasEmail = hasEmail;

this.hasAddress = hasAddress;

}

public String getName() {

return name;

}

public String getCategory() {

return category;

}

public String getPhoneNumber() {

return phoneNumber;

}

public boolean hasEmail() {

return hasEmail;

}

public boolean hasAddress() {

return hasAddress;

}

}

}