```java
package rolodex;

import javax.swing.*;
import java.awt.*;
import mvp.*;

/****
 *
 * Class RolodexMenuUI is a companion view to the <a href = "RolodexTool.html">
 * RolodexTool </a> model class.  The main component of the RolodexMenuUI
 * window is a JFrame with a menubar on the top and the data entry area below.
 * The contents of the data entry area vary depending on the chosen rolodex
 * operation.  To hold the varying contents, the data entry area is represented
 * as a JLayeredPane, which is an Java widget that holds components stacked one
 * on top of each other, with only the topmost component visible and active at
 * any given time.
 *                                                                          <p>
 * For an Add operation, the data entry area contains five JTextFields for
 * entering the values of the five card fields.  A JTextField is a Java widget
 * for entering single-line text values.  For the Delete, Change, and Find
 * operations, the data entry area contains a single JTextField for entering
 * the name of a card(s) to be deleted, changed, or found.
 *                                                                          <p>
 * Above the JTextFields is an operation-specific JLabel containing a message
 * describing the action of the operation.  A JLabel is a Java widget that
 * displays a static, borderless string of text in a selectable font.  For the
 * Add operation the message is "Enter Information for a Rolodex Card".  For
 * the other operations, the messages are similar.
 *                                                                          <p>
 * Below the text field(s) are operation-specific JButtons.  A JButton is a
 * Java widget that performs an action when pressed.  For the Add operation
 * there are OK, Clear, and Cancel buttons.  For the Delete and Change
 * operations there are OK and Cancel buttons.  For Delete, Change, and Find,
 * there are Next and Previous buttons in the case that the name entered in the
 * text field does not specify a unique card.
 *                                                                          <p>
 * For illustrative comparison, an alternative button-style GUI is defined in
 * class RolodexButtonUI in ./RolodexButtonUI.java, q.v.
 *
 */

public class RolodexMenuUI extends View {

    /**
     * Construct this with the given UI screen and RolodexTool model.  Also
     * construct the FileMenu, EditMenu, and RolodexMenu subviews.
     */
    public RolodexMenuUI(Screen s, RolodexTool rt) {

        /*
         * Call the parent constructor.
         */
        super(s, rt);

        /*
         * Construct the menubar.
         */
        jmb = new JMenuBar();

        /*
         * Construct the file menu, passing its File model.
         */
        fm = new FileMenu("File", rt.getFile());

        /*
         * Construct the edit menu, passing its Edit model.
         */
        em = new EditMenu("Edit", rt.getEdit());

        /*
         * Construct the rolodex menu, passing its Edit model.
         */
        rm = new RolodexMenu(new String("Rolodex"), rt.getRolodex(), this);

        /*
         * Construct the dialogs.
         */
        constructDialogs(rt.getRolodex());

        /*
         * For convenience, save a copy of the model as a downcast value.
         * Hence, this.m == this.rt, where the former is of type Model and
         * the latter is of type RolodexTool.
         */
        this.rt = rt;
    }

    /**
     * Construct the dialogs to be displayed in the data area.
     */
    protected void constructDialogs(Rolodex r) {
        acd = new AddCardDialog(r, this);
        dcd = new DeleteCardDialog(r, this);
        cncd = new ConfirmChangeDialog(r, this);
        ccd = new ChangeCardDialog(r, this, cncd);
        fcsd = new FoundCardsDialog(r, this);
        fcd = new FindCardDialog(r, this, fcsd);
        nfcsd = new NoFoundCardsDialog(r, this);
        emd = new ErrorMessageDialog(this);
        ed = new EmptyDialog();
    }

    /**
     * Compose the window of this as the top-level rolodex menubar.
     */
    public void compose() {

        /*
         * Make a new window for this, which in Java will be a JFrame -- the
         * standard outermost container for a Swing window.
         */
        window = new mvp.Window();
```

```java

        /*
         * Compose the menubar and set it as the menubar of this' window using
         * the JFrame.setMenuBar method.
         */
        window.setJMenuBar(composeMenuBar());

        /*
         * Compose the data entry area and insert into the content pane of the
         * window.  The content pane of a JFrame is the physical component into
         * which frame contents are added.
         */
        window.getContentPane().add(composeDataArea());

        /*
         * Make the start-up contents of the data entry empty.
         */
        showEmptyDialog();

        /*
         * Set the name of the top-level interactor, which name will appear in
         * the titlebar of the window.
         */
        window.setTitle("Rolodex Tool");

        /*
         * Call JFrame.pack to have Java size up the window properly.
         */
        window.pack();
    }

    /**
     * Display the add card dialog in the data entry area.
     */
    public void showAddCardDialog() {
        cur_dialog = acd;
        jlp.moveToFront(acd.getWidget());
    }

    /**
     * Display the delete card dialog in the data entry area.
     */
    public void showDeleteCardDialog() {
        cur_dialog = dcd;
        jlp.moveToFront(dcd.getWidget());
    }

    /**
     * Display the change card dialog in the data entry area.
     */
    public void showChangeCardDialog() {
        cur_dialog = ccd;
        jlp.moveToFront(ccd.getWidget());
    }

    /**
     * Display the confirm change card dialog in the data entry area.
     */
    public void showConfirmChangeDialog() {
        cur_dialog = cncd;
        jlp.moveToFront(cncd.getWidget());
    }

    /**
     * Display the find card dialog in the data entry area.
     */
    public void showFindCardDialog() {
        cur_dialog = fcd;
        jlp.moveToFront(fcd.getWidget());
    }

    /**
     * Display the found cards dialog in the data entry area.
     */
    public void showFoundDialog() {
        cur_dialog = fcsd;
        jlp.moveToFront(fcsd.getWidget());
    }

    /**
     * Display the no found cards dialog in the data entry area.
     */
    public void showNoFoundDialog() {
        cur_dialog = nfcsd;
        jlp.moveToFront(nfcsd.getWidget());
    }

    /**
     * Display the error message dialog in the data area with the given list of
     * strings as the error messages.  The list can be up to five messages
     * long, with each message shown in the order it appears in the list.
     * Before the error dialog is shown, this.prev_dialog is set to
     * this.cur_dialog so that the pre-error dialog can be restored after the
     * error dialog is dismissed by the user.
     */
    public void showErrorDialog(StringList sl) {
        emd.clearMessages();
        emd.setMessages(sl);
        prev_dialog = cur_dialog;
        jlp.moveToFront(emd.getWidget());
    }

    /**
     * Single-message overload of <a href="#showErrorDialog(StringList)">
     * showErrorDialog(StringList)</a>, q.v.
     */
    public void showErrorDialog(String s) {
        StringList sl = new StringList();
        sl.add(s);
        showErrorDialog(sl);
    }

```

```java
    /**
     * Display emptiness in the data entry area.
     */
    public void showEmptyDialog() {
        jlp.moveToFront(ed.getWidget());
    }

    /**
     * Return this’ rolodex menu.
     */
    RolodexMenu getRolodexMenu() {
        return rm;
    }


    /**
     * Restore the dialog that was displayed immediately before the error
     * dialog was shown.
     */
    protected void restorePreErrorDialog() {
        cur_dialog = prev_dialog;
        jlp.moveToFront(cur_dialog.getWidget());
    }

    /*-*
     * Component composition functions.
     */

    /**
     * Compose and return the menubar.
     */
    protected JMenuBar composeMenuBar() {
        jmb.add(composeFileMenu().getMenu());
        jmb.add(composeEditMenu().getMenu());
        jmb.add(composeRolodexMenu().getMenu());
        return jmb;
    }

    /**
     * Compose and return the file pulldown menu.
     */
    protected FileMenu composeFileMenu() {
        fm.compose();
        return fm;
    }

    /**
     * Compose and return the edit pulldown menu.
     */
    protected EditMenu composeEditMenu() {
        em.compose();
        return em;
    }

    /**
     * Compose and return the rolodex pulldown menu.
     */
    protected RolodexMenu composeRolodexMenu() {
        rm.compose();
        return rm;
    }

    /**
     * Compose the data entry/display area below the menubar.  Per the the
     * specs, it is fixed size, big enough to hold the largest of the operation
     * dialogs.
     */
    protected JLayeredPane composeDataArea() {

        /*
         * Create a new layered pane.
         */
        jlp = new JLayeredPane();

        /*
         * Set the pane’s preferred size.  Per the Java documentation, layered
         * panes must have a preferred size because they do not have a default
         * layout manager.
         */
        jlp.setPreferredSize(new Dimension(DATA_AREA_WIDTH, DATA_AREA_HEIGHT));

        /*
         * Compose each of the rolodex dialogs, e.g., for adding a card,
         * deleting a card, etc.  Each dialog will be added to the layered pane
         * as a new layer.
         */
        composeDialogs();

        /*
         * Returned the fully composed layered pane.
         */
        return jlp;
    }

    /**
     * Compose the UI dialog boxes for adding, deleting, finding, and changing
     * a card in the rolodex.  Also compose the display for found card output
     * that lists the cards found.  Each of these dialogs appears in the data
     * entry/display area.
     */
    protected void composeDialogs() {
        /*
         * Add
         */
        composeAndInsert(acd);

        /*
         * Delete
         */
        composeAndInsert(dcd);

        /*
```

```
         * Change
         */
        composeAndInsert(ccd);

        /*
         * Confirm Change
         */
        composeAndInsert(cncd);

        /*
         * Found.  (Note: this is seemingly done out of order since
         * FindCardDialog needs a pointer to FoundCardsDialog.  But then the
         * order of the cards is really irrelevant anyway, except for
         * organizational clarity.)
         */
         composeAndInsert(fcsd);

        /*
         * Find
         */
         composeAndInsert(fcd);

        /*
         * No Cards Found
         */
        composeAndInsert(nfcsd);

        /*
         * Error
         */
        composeAndInsert(emd);

        /*
         * Empty
         */
        composeAndInsert(ed);
    }

    /**
     * Compose the given view, which will be one of the data area dialogs.  Set
     * the size of the view's widget and add it into the layered pane.  Note
     * that the explicit size setting is necessary for the components of a
     * layered pane.
     */
    protected void composeAndInsert(View v) {
        v.compose();
        v.getWidget().setBounds(0, 0, DATA_AREA_WIDTH, DATA_AREA_HEIGHT);
        jlp.add(v.getWidget());
    }


    /*-*
     * Data members
     */

    /** Width of data area.  This explicit size value is necessary since the
     * components of a layered pane must be explicitly dimensioned. */
    public static final int DATA_AREA_WIDTH = 400;

    /** Height of data area.  This explicit size value is necessary since the
     * components of a layered pane must be explicitly dimensioned. */
    public static final int DATA_AREA_HEIGHT = 300;

    /** Font name for all dialog labels. */
    public static String LABEL_FONT_NAME = "Times";

    /** Companion model, downcast version of this.model */
    protected RolodexTool rt;

    /** Pulldown menubar */
    protected JMenuBar jmb;

    /** File pulldown menu */
    protected FileMenu fm;

    /** Edit pulldown menu */
    protected EditMenu em;

    /** Rolodex pulldown menu */
    protected RolodexMenu rm;

    /** JLayeredPane holding operation-specific dialogs */
    protected JLayeredPane jlp;

    /** Add card dialog */
    protected AddCardDialog acd;

    /** Delete card dialog */
    protected DeleteCardDialog dcd;

    /** Change card dialog */
    protected ChangeCardDialog ccd;

    /** Confirm change dialog */
    protected ConfirmChangeDialog cncd;

    /** Find card dialog */
    protected FindCardDialog fcd;

    /** Found card(s) dialog */
    protected FoundCardsDialog fcsd;

    /** No found cards dialog */
    protected NoFoundCardsDialog nfcsd;

    /** General error messaging dialog */
    protected ErrorMessageDialog emd;

    /** Empty dialog */
    protected EmptyDialog ed;

    /** Current dialog on the top of panel */
```

```
    protected View cur_dialog;

    /** Pre-error dialog for restoration after error message */
    protected View prev_dialog;

}
```