```
Loading vc-cvs...
package caltool.view_ui;

import caltool.view.*;
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
// import mvp.*;        // Not imported to avoid conflict with caltool.View

/****
 *
 * Class ViewMenu is the pulldown menu view of the <a href = "View.html" View
 * </a> model class.  The ViewMenu widget is a Java JMenu.  Anonymous instances
 * of JMenuItem are defined for each item in the menu.
 *
 */
public class ViewMenu extends mvp.View {

    /**
     * Construct this with the given name as the pulldown label.  The given
     * View is the companion model.  Also construct the state-changing menu
     * items, that must be persistent so they're text can change.
     */
    public ViewMenu(mvp.Screen screen, View view, ViewUI viewUI) {
        super(screen, view);
        this.viewUI = viewUI;

        showHideAppointmentsItem = new JMenuItem("Hide Appointments");
        showHideMeetingsItem = new JMenuItem("Hide Meetings");
        showHideTasksItem = new JMenuItem("Hide Tasks");
        showHideEventsItem = new JMenuItem("Hide Events");
    }

    /**
     * Compose this by inserting each of its menu items into the pulldown menu.
     * The items are: Item, Day, Week, Month, Year, Next, Previous, Today, Goto
     * Date, Lists, Filter, Other User, Group, Windows, and Calendars.
     * Separators are placed after the Year, Goto Date, Filter, and Group
     * items.
     *
     * A menu item is created with the following general code pattern:
     *                                                                 <pre>
     *     JMenu.add(new JMenuItem("<em>Item name</em>").addActionListener(
     *         new ActionListener() {
     *             public void actionPerformed(ActionEvent e) {
     *                 <em>model.method()</em>
     *                                                                 </pre>
     */
    public Component compose() {

        /*
         * Make the widget of this a JMenu.
         */
        widget = menu = new JMenu("View");

        /*
         * Add the first group of items -- Item through year.
         */
        addItemItem();
        addDayItem();
        addWeekSubmenu();
        addMonthItem();
        addYearItem();
        menu.add(new JSeparator());

        /*
         * Add the next group of items -- Next through Goto Date.
         */
        addNextItem();
        addPreviousItem();
        addTodayItem();
        addGotoDateItem();
        menu.add(new JSeparator());

        /*
         * Add the Lists and Filter submenus.
         */
        addListsSubmenu();
        addFilterSubmenu();
        menu.add(new JSeparator());

        /*
         * Add the Other User and Group items.
         */
        addOtherUserItem();
        addGroupItem();
        menu.add(new JSeparator());

        /*
         * Add Windows and Calendars submenus.
         */
        addWindowsSubmenu();
        addCalendarsSubmenu();

        return widget;
    }

    /**
     * Add the 'Item' menu item.
     */
    public void addItemItem() {
        menu.add(new JMenuItem("Item")).addActionListener(
            new ActionListener() {
                public void actionPerformed(ActionEvent ev) {
                    viewUI.getItemEditor().update(null,
                        ((View)model).viewItem());
                    viewUI.getItemEditor().show();
                }

            }
        );
    }
```

```java

    /**
     * Add the 'Day' menu item.
     */
    public void addDayItem() {
        menu.add(new JMenuItem("Day")).addActionListener(
            new ActionListener() {
                public void actionPerformed(ActionEvent ev) {
                    System.out.println("In View.viewDay.");
                }

            }
        );
    }

    /**
     * Add the 'Week' menu item, which is a cascading pullright containing
     * 'Table' and 'List' items.
     */
    public void addWeekSubmenu() {
        submenu = new JMenu("Week");
        menu.add(submenu);

        /*
         * Add the 'Week->Table' menu item.
         */
        submenu.add(new JMenuItem("Table")).addActionListener(
            new ActionListener() {
                public void actionPerformed(ActionEvent ev) {
                    System.out.println("In View.viewWeekTable.");
                }

            }
        );


        /*
         * Add the 'Week->Lists' menu item.
         */
        submenu.add(new JMenuItem("Lists")).addActionListener(
            new ActionListener() {
                public void actionPerformed(ActionEvent ev) {
                    System.out.println("In View.viewWeekLists.");
                }

            }
        );

    }

    /*
     * Add the 'Month' menu item.
     */
    public void addMonthItem() {
        menu.add(new JMenuItem("Month")).addActionListener(
            new ActionListener() {
                public void actionPerformed(ActionEvent ev) {
                    JFrame jf = new JFrame();
                    jf.getContentPane().add(new JLabel("xxx"));
                    jf.setVisible(true);
                    ((View)model).viewMonth();
                    viewUI.getMonthlyAgendaDisplay().update(null, null);
                    viewUI.getMonthlyAgendaDisplay().show();
                }
            }
        );
    }

    /**
     * Add the 'Year' menu item.
     */
    public void addYearItem() {
        menu.add(new JMenuItem("Year")).addActionListener(
            new ActionListener() {
                public void actionPerformed(ActionEvent ev) {
                    System.out.println("In View.viewYear.");
                }
            }
        );
    }

    /**
     * Add the 'Next' menu item.
     */
    public void addNextItem() {
        menu.add(new JMenuItem("Next")).addActionListener(
            new ActionListener() {
                public void actionPerformed(ActionEvent ev) {
                    System.out.println("In View.gotoNext.");
                }
            }
        );
    }

    /**
     * Add the 'Previous' menu item.
     */
    public void addPreviousItem() {
        menu.add(new JMenuItem("Previous")).addActionListener(
            new ActionListener() {
                public void actionPerformed(ActionEvent ev) {
                    System.out.println("In View.gotoPrevious.");
                }
            }
        );
    }

    /**
     * Add the 'Today' menu item.
     */
    public void addTodayItem() {
        menu.add(new JMenuItem("Today")).addActionListener(
```

```
            new ActionListener() {
                public void actionPerformed(ActionEvent ev) {
                    System.out.println("In View.gotoToday.");
                }
            }
        );
    }

    /**
     * Add the 'Goto Date ...' menu item.
     */
    public void addGotoDateItem() {
        menu.add(new JMenuItem("Goto Date ...")).addActionListener(
            new ActionListener() {
                public void actionPerformed(ActionEvent ev) {
                    System.out.println("In View.gotoDate.");
                }
            }
        );
    }

    /**
     * Add the 'Lists' menu item, which is a cascading pullright containing
     * 'Appointments', 'Meetings', 'Tasks', 'Events', 'All Items', and
     * 'Custom' items.
     */
    public void addListsSubmenu() {
        submenu = new JMenu("Lists");
        menu.add(submenu);

        /*
         * Add the top five items.
         */
        addListsAppointmentsItem();
        addListsMeetingsItem();
        addListsTasksItem();
        addListsEventsItem();
        addListsAllItemsItem();
        submenu.add(new JSeparator());

        /*
         * Add the Custom submenu.
         */
        addListsCustomSubmenu();
    }

    /**
     * Add the 'Lists->Appointments' menu item.
     */
    public void addListsAppointmentsItem() {
        submenu.add(new JMenuItem("Appointments")).addActionListener(
            new ActionListener() {
                public void actionPerformed(ActionEvent ev) {
                    viewUI.getAppointmentsListDisplay().update(null, null);
                    viewUI.getAppointmentsListDisplay().show();
                }

            }
        );
    }

    /**
     * Add the 'Lists->Meetings' menu item.
     */
    public void addListsMeetingsItem() {
        submenu.add(new JMenuItem("Meetings")).addActionListener(
            new ActionListener() {
                public void actionPerformed(ActionEvent ev) {
                    System.out.println("In View.viewListsMeetings.");
                }

            }
        );
    }

    /**
     * Add the 'Lists->Tasks' menu item.
     */
    public void addListsTasksItem() {
        submenu.add(new JMenuItem("Tasks")).addActionListener(
            new ActionListener() {
                public void actionPerformed(ActionEvent ev) {
                    System.out.println("In View.viewListsTasks.");
                }

            }
        );
    }

    /**
     * Add the 'Lists->Events' menu item.
     */
    public void addListsEventsItem() {
        submenu.add(new JMenuItem("Events")).addActionListener(
            new ActionListener() {
                public void actionPerformed(ActionEvent ev) {
                    System.out.println("In View.viewListsEvents.");
                }

            }
        );
    }

    /**
     * Add the 'Lists->All Items' menu item.
     */
    public void addListsAllItemsItem() {
        submenu.add(new JMenuItem("All Items")).addActionListener(
            new ActionListener() {
                public void actionPerformed(ActionEvent ev) {
                    System.out.println("In View.viewListsAllItems.");
                }
```

```java

            }
        );
    }

    /**
     * Add the 'Lists->Custom' submenu.
     */
    public void addListsCustomSubmenu() {
        submenu2 = new JMenu("Custom");
        submenu.add(submenu2);

        /*
         * Add a separator.
         */
        submenu2.add(new JSeparator());

        /*
         * Add the 'Lists->Custom->Edit ...' menu item.
         */
        submenu2.add(new JMenuItem("Edit ...")).addActionListener(
            new ActionListener() {
                public void actionPerformed(ActionEvent ev) {
                    System.out.println("In View.viewListsCustomEdit.");
                }

            }
        );
    }

    /**
     * Add the 'Filter' menu item, which is a cascading pullright
     * containing 'Show/Hide Appointments', 'Show/Hide Meetings',
     * 'Show/Hide Tasks', 'Show/Hide Events', and 'Custom' items.
     */
    public void addFilterSubmenu() {
        submenu = new JMenu("Filter");
        menu.add(submenu);

        /*
         * Add the top four items.
         */
        addFilterAppointmentsItem();
        addFilterMeetingsItem();
        addFilterTasksItem();
        addFilterEventsItem();
        submenu.add(new JSeparator());

        /*
         * Add the Custom submenu.
         */
        addFilterCustomSubmenu();
    }

    /**
     * Add the 'Filter->Show/Hide Appointments' menu item.
     */
    public void addFilterAppointmentsItem() {
        submenu.add(showHideAppointmentsItem).addActionListener(
            new ActionListener() {
                public void actionPerformed(ActionEvent ev) {
                    ((View)model).toggleShowHideAppointments();
                    showHideAppointmentsItem.setText(
                        (((View)model).areAppointmentsHidden()
                            ? "Show" : "Hide") + " Appointments");
                }
            }
        );
    }

    /**
     * Add the 'Filter->Show/Hide Meetings' menu item.
     */
    public void addFilterMeetingsItem() {
        submenu.add(showHideMeetingsItem).addActionListener(
            new ActionListener() {
                public void actionPerformed(ActionEvent ev) {
                    ((View)model).toggleShowHideMeetings();
                    showHideMeetingsItem.setText(
                        (((View)model).areMeetingsHidden()
                            ? "Show" : "Hide") + " Meetings");
                }

            }
        );
    }

    /**
     * Add the 'Filter->Show/Hide Tasks' menu item.
     */
    public void addFilterTasksItem() {
        submenu.add(showHideTasksItem).addActionListener(
            new ActionListener() {
                public void actionPerformed(ActionEvent ev) {
                    ((View)model).toggleShowHideTasks();
                    showHideTasksItem.setText(
                        (((View)model).areTasksHidden()
                            ? "Show" : "Hide") + " Tasks");
                }

            }
        );
    }

    /**
     * Add the 'Filter->Show/Hide Events' menu item.
     */
    public void addFilterEventsItem() {
        submenu.add(showHideEventsItem).addActionListener(
            new ActionListener() {
                public void actionPerformed(ActionEvent ev) {
                    ((View)model).toggleShowHideEvents();
```

```
                    showHideEventsItem.setText(
                        (((View)model).areEventsHidden()
                            ? "Show" : "Hide") + " Events");
                }

            }
        );
    }

    /**
     * Add the 'Filter->Custom' submenu.
     */
    public void addFilterCustomSubmenu() {
        submenu2 = new JMenu("Custom");
        submenu.add(submenu2);

        /*
         * Add a separator.
         */
        submenu2.add(new JSeparator());

        /*
         * Add the 'Filter->Custom->Edit ...' menu item.
         */
        submenu2.add(new JMenuItem("Edit ...")).addActionListener(
            new ActionListener() {
                public void actionPerformed(ActionEvent ev) {
                    System.out.println("In View.viewFilterCustomEdit.");
                }

            }
        );
    }

    /**
     * Add the 'Other User ...' menu item.
     */
    public void addOtherUserItem() {
        menu.add(new JMenuItem("Other User ...")).addActionListener(
            new ActionListener() {
                public void actionPerformed(ActionEvent ev) {
                    System.out.println("In View.viewOtherUser.");
                }
            }
        );
    }

    /**
     * Add the 'Group ...' menu item.
     */
    public void addGroupItem() {
        menu.add(new JMenuItem("Group ...")).addActionListener(
            new ActionListener() {
                public void actionPerformed(ActionEvent ev) {
                    System.out.println("In View.viewGroup.");
                }
            }
        );
    }

    /**
     * Add the 'Windows' menu item.
     */
    public void addWindowsSubmenu() {
        submenu = new JMenu("Windows");
        menu.add(submenu);

        /*
         * Add the Close item with a separator above and below.
         */
        submenu.add(new JSeparator());
        addWindowsCloseItem();
        submenu.add(new JSeparator());

        /*
         * Add the Windowing Mode submenu and Magnetize item.
         */
        addWindowsWindowingModeSubmenu();
        addWindowsMagnetizeItem();
    }

    /**
     * Add the 'Windows->Close' menu item.
     */
    public void addWindowsCloseItem() {
        submenu.add(new JMenuItem("Close")).addActionListener(
            new ActionListener() {
                public void actionPerformed(ActionEvent ev) {
                    System.out.println("In View.close.");
                }
            }
        );
    }

    /**
     * Add the 'Windows->Windowing Mode' submenu.
     */
    public void addWindowsWindowingModeSubmenu() {
        submenu2 = new JMenu("Windowing Mode");
        submenu.add(submenu2);
        group = new ButtonGroup();

        /*
         * Add the three radio button items.
         */
        addWindowsTwoWindowModeItem();
        addWindowsPerLevelModeItem();
        addWindowsMultiWindowModeItem();
    }


    /**
```

```java
     * Add the 'Windows->Windowing Mode->Two-Window' menu item.
     */
    public void addWindowsTwoWindowModeItem() {
        submenu2.add(rbItem =
            new JRadioButtonMenuItem("Two-Window")).addActionListener(
                new ActionListener() {
                    public void actionPerformed(ActionEvent ev) {
                        System.out.println("In View.setTwoWindowMode.");
                    }
                }
        );

        group.add(rbItem);
    }

    /**
     * Add the 'Windows->Windowing Mode->Per-Level' menu item.
     */
    public void addWindowsPerLevelModeItem() {
        submenu2.add(rbItem =
            new JRadioButtonMenuItem("Per-Level")).addActionListener(
                new ActionListener() {
                    public void actionPerformed(ActionEvent ev) {
                        System.out.println("In View.setPerLevelWindowMode.");
                    }
                }
            );

        rbItem.setSelected(true);
        group.add(rbItem);
    }

    /**
     * Add the 'Windows->Windowing Mode->Multi-Window' menu item.
     */
    public void addWindowsMultiWindowModeItem() {
        submenu2.add(rbItem =
          new JRadioButtonMenuItem("Multi-Window")).addActionListener(
            new ActionListener() {
                public void actionPerformed(ActionEvent ev) {
                    System.out.println("In View.setMultiWindowMode.");
                }
            }
        );

        group.add(rbItem);
    }

    /**
     * Add the 'Windows->Magnetize' menu item.
     */
    public void addWindowsMagnetizeItem() {
        submenu.add(new JMenuItem("Magnetize")).addActionListener(
            new ActionListener() {
                public void actionPerformed(ActionEvent ev) {
                    System.out.println("In View.magnetize.");
                }
            }
        );
    }

    /**
     * Add the 'Calendars' menu item.
     */
    public void addCalendarsSubmenu() {
        submenu = new JMenu("Calendars");
        menu.add(submenu);

        /*
         * This submenu will be populated by the chronologically-ordered list
         * of all open calendars.
         */

    }


    /** Temp submenu vars, used in item-adding methods */
    protected JMenu submenu, submenu2;

    /** Pre-cast reference to this' widget, which is a menu */
    protected JMenu menu;

    /** Button group for radio button menu items */
    ButtonGroup group;

    /** Temp radio button menu item */
    JRadioButtonMenuItem rbItem;

    /** Show/Hide Appointments menu item.  It's a persistent data field so it's
        text can change. */
    protected JMenuItem showHideAppointmentsItem;


    /** Show/Hide Meetings menu item.  It's a persistent data field so it's
        text can change. */
    protected JMenuItem showHideMeetingsItem;


    /** Show/Hide Tasks menu item.  It's a persistent data field so it's
        text can change. */
    protected JMenuItem showHideTasksItem;


    /** Show/Hide Events menu item.  It's a persistent data field so it's
        text can change. */
    protected JMenuItem showHideEventsItem;

    /** The parent view. */
    ViewUI viewUI;

}
```