```
Loading vc-cvs...
```

```
   1  package caltool.view_ui;
   2
   3  import caltool.schedule.*;
   4  import mvp.*;
   5  import java.util.*;
   6
   7  /****
   8   *
   9   * Class ItemEditor is a dispatching class for the currently selected item.
  10   * The update and show methods ask the workspace for the currently selected
  11   * item, then dispatch to the update and show methods of the appropriate
  12   * item-editor subclasss, namely one of AppointmentEditor, MeetingEditor,
  13   * TaskEditor, or EventEditor.
  14   *                                                                      <p>
  15   * There's no particularly elegant way to avoid the explicit dispatch via
  16   * dyanic binding, since the dispatch is to view classes, not model classes.
  17   * In order to use dynamic-binding dispatch from an instance of ScheduledItem,
  18   * we would have to implement update and show methods in the model classes,
  19   * which goes against the style of view-ignorant model design we're using.
  20   *
  21   */
  22
  23  public class ItemEditor extends View {
  24
  25      public ItemEditor(ViewUI viewUI) {
  26          this.viewUI = viewUI;
  27          editor = null;
  28      }
  29
  30      public void update(Observable o, Object arg) {
  31
  32          if (arg.getClass() == Appointment.class)
  33              editor = viewUI.getAppointmentEditor();
  34          else if (arg.getClass() == Meeting.class)
  35              editor = viewUI.getMeetingEditor();
  36          else if (arg.getClass() == Task.class)
  37              editor = viewUI.getTaskEditor();
  38          else if (arg.getClass() == Event.class)
  39              editor = viewUI.getTaskEditor();
  40
  41          editor.update(o, arg);
  42
  43      }
  44
  45      public void show() {
  46          if (editor != null) {
  47              editor.show();
  48          }
  49      }
  50
  51      View editor;
  52      ViewUI viewUI;
  53
  54  }
```