```
/****
 *
 * Implementation of rolodex-menu-ui.h
 *
 */

#include "edit-menu.h"
#include "file-menu.h"
#include "rolodex-menu-ui.h"
#include "rolodex-menu.h"
#include "rolodex-tool.h"
#include "white-background-tray.h"
#include <InterViews/border.h>
#include <InterViews/glue.h>
#include <InterViews/message.h>
#include <InterViews/tray.h>


RolodexMenuUI::RolodexMenuUI(Screen* s, RolodexTool* rt) :
        RolodexUI(s, rt) {
    /*
     * Construct the file menu, passing its File model.
     */
    fm = new FileMenu(new String("File"), rt->GetFile());

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

    /*
     * Construct the edit menu, passing its Edit model.
     */
    rm = new RolodexMenu(new String("Rolodex"), rt->GetRolodex(), this);

    /*
     * 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;
}

RolodexMenuUI::~RolodexMenuUI() {
}

void RolodexMenuUI::Compose() {

    /*
     * Make the window of this a WhiteBackgroundTray, which isspecialization of
     * an InterViews Tray.  A Tray is a standard outermost interactor for a
     * top-level interface since it provides a opaque background.  An HBox or
     * VBox by itself has a transparent background, which means that if it's
     * used as the outmost interactor, it will "absorb" whatever window
     * background is underneath it.  Hence, it is standard practice to use a
     * Tray for the solid background.  Trays have other useful purposes, but
     * here we're simply using one for its background.
     */
    w = new WhiteBackgroundTray(vb = new VBox());

    /*
     * Compose the menubar and insert it into this' window.  The border
     * provides a visible separator under the menubar.  By default a menubar
     * has no visible border.
     */
    vb->Insert(ComposeMenuBar());
    vb->Insert(new HBorder());

    /*
     * Compose the data entry area and insert into the window.
     */
    vb->Insert(ComposeDataArea());

    /*
     * Make the start-up contents of the data entry empty.
     */
    d->FlipTo(EMPTYPOSITION);
}

MenuBar* RolodexMenuUI::ComposeMenuBar() {
    mb = new MenuBar();

    /*
     * Compose the file menu view and insert its window into the bar.
     */
    mb->Include((PulldownMenu*)ComposeFileMenu()->GetWindow());

    /*
     * Stick some glue between the file and edit menus.
     */
    mb->Insert(new HGlue(
        round(0.2*inch), round(0.15*inch), 0));
     // ^^^^^^^^^^^^^^^^  ^^^^^^^^^^^^^^^^  ^ Stretchibilty,
     // ^^^^^^^^^^^^^^^^  ^^^^^^^^^^^^^^^^
     // ^^^^^^^^^^^^^^^^  ^^^^^^^^^^^^^^^^ Shrinkibility, i.e., minimum size
     // ^^^^^^^^^^^^^^^^
     // ^^^^^^^^^^^^^^^^ Natural size, i.e., initial size

    /*
     * Compose the edit pulldown and insert it into the bar.
     */
    mb->Include((PulldownMenu*)ComposeEditMenu()->GetWindow());

    /*
     * Stick some glue between the edit and rolodex menus.
     */
    mb->Insert(new HGlue(round(0.2*inch), round(0.15*inch), 0));

    /*
     * Compose the rolodex pulldown and insert it into the bar.
     */
    mb->Include((PulldownMenu*)ComposeRolodexMenu()->GetWindow());
```

```

    /*
     * Stick some glue on the right edge of the menubar.
     */
    mb->Insert(new HGlue(round(0.15*inch), 0, hfil));

    return mb;
}

HBox* RolodexMenuUI::ComposeDataArea() {

    /*
     * Allocate a deck to hold the operation-specific dialogs.
     */
    d = new Deck();

    /*
     * Allocate and comppose each of the dialogs and insert each into the deck.
     * Note that the model for each of the dialogs is an appropriate components
     * of the rolodex tool workspace
     */
    ComposeDialogs();

    /*
     * Put the deck in an HBox with some glue on each end.  This outer hbox
     * with its glue will allow horizontal stretchibility.  Note here how the
     * HBox constructor can take more than one argument -- it can take up to 7
     * arguments.
     */
    return new HBox(
        new HGlue(round(0.2 * inch), 0, hfil),
        d,
        new HGlue(round(0.2 * inch), 0, hfil)
    );
}

FileMenu* RolodexMenuUI::ComposeFileMenu() {

    /*
     * Compose the file menu, which entails inserting each of its items.
     */
    fm->Compose();

    /*
     * Return the composed menu.
     */
    return fm;
}

EditMenu* RolodexMenuUI::ComposeEditMenu() {

    /*
     * Compose the edit menu, which entails inserting each of its items.
     */
    em->Compose();

    /*
     * Return the composed menu.
     */
    return em;
}

RolodexMenu* RolodexMenuUI::ComposeRolodexMenu() {

    /*
     * Compose the rolodex menu, which entails inserting each of its items.
     */
    rm->Compose();

    /*
     * Return the composed menu.
     */
    return rm;
}

void RolodexMenuUI::ComposeDialogs() {

    /*
     * Position 1 -- Add.
     */
    acd = new AddCardDialog(rt->GetRolodex(), this);
    ComposeAndInsert(acd);

    /*
     * Position 2 -- Delete.
     */
    dcd = new DeleteCardDialog(rt->GetRolodex(), this);
    ComposeAndInsert(dcd);

    /*
     * Position 3 -- Confirm Delete.
     */
    cdd = new ConfirmDeleteDialog(this);
    ComposeAndInsert(cdd);

    /*
     * Position 4 -- Change.
     */
    ccd = new ChangeCardDialog();
    ComposeAndInsert(ccd);

    /*
     * Position 5 -- Confirm Change.
     */
    cncd = new ConfirmChangeDialog(rt->GetRolodex(), this);
    ComposeAndInsert(cncd);

    /*
     * Position 6 -- Find.
     */
    fcd = new FindCardDialog(rt->GetRolodex(), this);
    ComposeAndInsert(fcd);
```

```

     /*
      * Position 7 -- Found.
      */
    fcsd = new FoundCardsDialog(this);
    ComposeAndInsert(fcsd);

     /*
      * Position 8 -- Error.
      */
    emd = new ErrorMessageDialog(this);
    ComposeAndInsert(emd);

     /*
      * Position 9 -- Empty.
      */
     d->Insert(new Tray());
}

void RolodexMenuUI::ComposeAndInsert(View* v) {
    v->Compose();
    d->Insert(v->GetWindow());
}

void RolodexMenuUI::ShowAddCardDialog() {
    d->FlipTo(ADDPOSITION);
}

void RolodexMenuUI::ShowDeleteCardDialog() {
    d->FlipTo(DELPOSITION);
}

void RolodexMenuUI::ShowChangeCardDialog() {
    d->FlipTo(CHANGEPOSITION);
}

void RolodexMenuUI::ShowFindCardDialog() {
    d->FlipTo(FINDPOSITION);
}

void RolodexMenuUI::ShowFoundDialog() {
    d->FlipTo(FOUNDPOSITION);
}

void RolodexMenuUI::ShowEmptyDialog() {
    d->FlipTo(EMPTYPOSITION);
}
```