```
/****
 *
 * Class EditMenu is the companion view to the Edit model class.  The EditMenu
 * window is an InterViews PullDownMenu.
 *
 * Classes NewMenuItem, OpenMenuItem, SaveMenuItem, SaveAsMenuItem,
 * PrintMenuItem, and QuitMenuItem specialize InterViews MenuItem to define the
 * items on the EditMenu window.
 *
 */

#ifndef editmenuIncluded
#define editmenuIncluded

#include "edit.h"
#include "view.h"
#include <InterViews/menu.h>


class EditMenu : public View {
  public:
    EditMenu(Name* n, Edit* e);
    /*
     * Construct this with the given name as the pulldown label.  The given
     * Edit is the companion model.
     */

    ~EditMenu();
    /*
     * Deep destruct this by deleting all of its items.
     */

    virtual void Compose();
    /*
     * Compose this by inserting each of its six menu items -- New, Open, Save,
     * SaveAs, Print, and Quit -- into the pulldown menu window.
     */

  protected:
    Edit* e;            // Companion model (downcast version of this->m)
    PulldownMenu* pm;   // For convenience, downcast version of this->w
};

class UndoMenuItem : public MenuItem {
  public:
    UndoMenuItem(Edit* e);
    /*
     * Construct this with the given Edit model.
     */

  protected:
    virtual void Do();
    /*
     * Respond to the selection of this menu item by calling the Undo model
     * function.
     */

    Edit* e;            // The compnion model
};

class RedoMenuItem : public MenuItem {
  public:
    RedoMenuItem(Edit* e);
    /*
     * Construct this with the given Edit model.
     */

  protected:
    virtual void Do();
    /*
     * Respond to the selection of this menu item by calling the Redo model
     * function.
     */

    Edit* e;            // The compnion model
};

class RepeatMenuItem : public MenuItem {
  public:
    RepeatMenuItem(Edit* e);
    /*
     * Construct this with the given Edit model.
     */

  protected:
    virtual void Do();
    /*
     * Respond to the selection of this menu item by calling the Repeat model
     * function.
     */

    Edit* e;            // The compnion model
};

class CutMenuItem : public MenuItem {
  public:
    CutMenuItem(Edit* e);
    /*
     * Construct this with the given Edit model.
     */

  protected:
    virtual void Do();
    /*
     * Respond to the selection of this menu item by calling the Cut model
     * function.
     */

    Edit* e;            // The compnion model
};

class CopyMenuItem : public MenuItem {
```

```
113   public:
114     CopyMenuItem(Edit* e);
115     /*
116      * Construct this with the given Edit model.
117      */
118
119   protected:
120     virtual void Do();
121     /*
122      * Respond to the selection of this menu item by calling the Copy model
123      * function.
124      */
125
126     Edit* e;            // The compnion model
127 };
128
129 class PasteMenuItem : public MenuItem {
130   public:
131     PasteMenuItem(Edit* e);
132     /*
133      * Construct this with the given Edit model.
134      */
135
136   protected:
137     virtual void Do();
138     /*
139      * Respond to the selection of this menu item by calling the Paste model
140      * function.
141      */
142
143     /**
144      * Data members.
145      */
146     Edit* e;            // The compnion model (downcast copy of this->m)
147     PulldownMenu* pm;   // The window (Downcast copy of this->w)
148
149 };
150
151 class EditDeleteMenuItem : public MenuItem {
152   public:
153     EditDeleteMenuItem(Edit* e);
154     /*
155      * Construct this with the given Edit model.
156      */
157
158   protected:
159     virtual void Do();
160     /*
161      * Respond to the selection of this menu item by calling the Delete model
162      * function.
163      */
164
165     /**
166      * Data members.
167      */
168     Edit* e;            // The compnion model (downcast copy of this->m)
169     PulldownMenu* pm;   // The window (Downcast copy of this->w)
170
171 };
172
173 class SelectAllMenuItem : public MenuItem {
174   public:
175     SelectAllMenuItem(Edit* e);
176     /*
177      * Construct this with the given Edit model.
178      */
179
180   protected:
181     virtual void Do();
182     /*
183      * Respond to the selection of this menu item by calling the SelectAll
184      * model function.
185      */
186
187     /**
188      * Data members.
189      */
190     Edit* e;            // The compnion model (downcast copy of this->m)
191     PulldownMenu* pm;   // The window (Downcast copy of this->w)
192
193 };
194
195 class FindMenuItem : public MenuItem {
196   public:
197     FindMenuItem(Edit* e);
198     /*
199      * Construct this with the given Edit model.
200      */
201
202   protected:
203     virtual void Do();
204     /*
205      * Respond to the selection of this menu item by calling the Find
206      * model function.
207      */
208
209     /**
210      * Data members.
211      */
212     Edit* e;            // The compnion model (downcast copy of this->m)
213     PulldownMenu* pm;   // The window (Downcast copy of this->w)
214
215 };
216
217 class CommandMenuItem : public MenuItem {
218   public:
219     CommandMenuItem(Edit* e);
220     /*
221      * Construct this with the given Edit model.
222      */
223
224   protected:
```

```
225      virtual void Do();
226      /*
227       * Respond to the selection of this menu item by calling the Command
228       * model function.
229       */
230
231      /**
232       * Data members.
233       */
234      Edit* e;            // The compnion model (downcast copy of this->m)
235      PulldownMenu* pm;   // The window (Downcast copy of this->w)
236
237  };
238
239  #endif
```