HomeBase

TextApplication v1.0

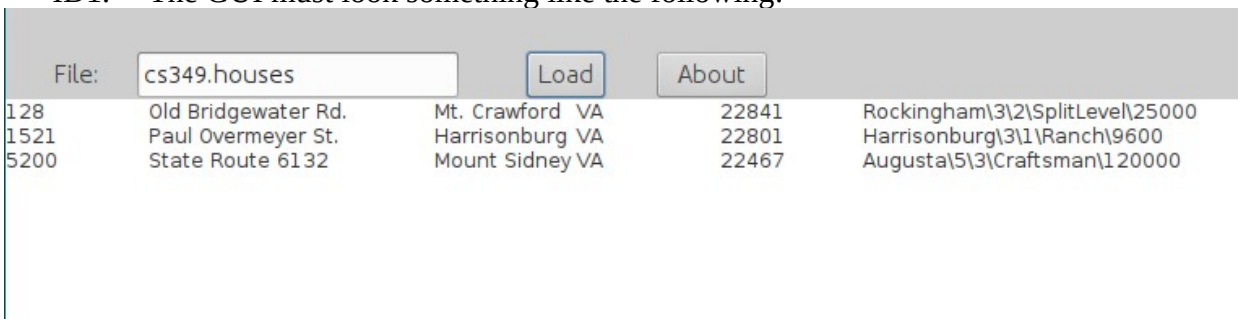
Overview

`TextApplication` is a simple GUI-based application that allows users to view either apartment listings or house listings.

Interaction Design

The system must satisfy the following interaction design requirements.

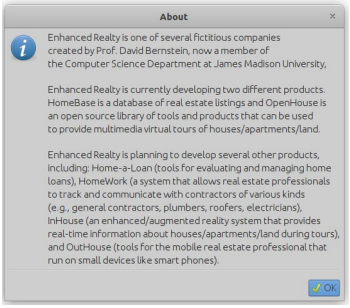
ID1. The GUI must look something like the following:



ID2. The user must be able to enter a file name into the text field labeled "File:".

ID3. Clicking on the Load button must cause the display to be cleared, the selected file to be read, and the information in that file shown in the display.

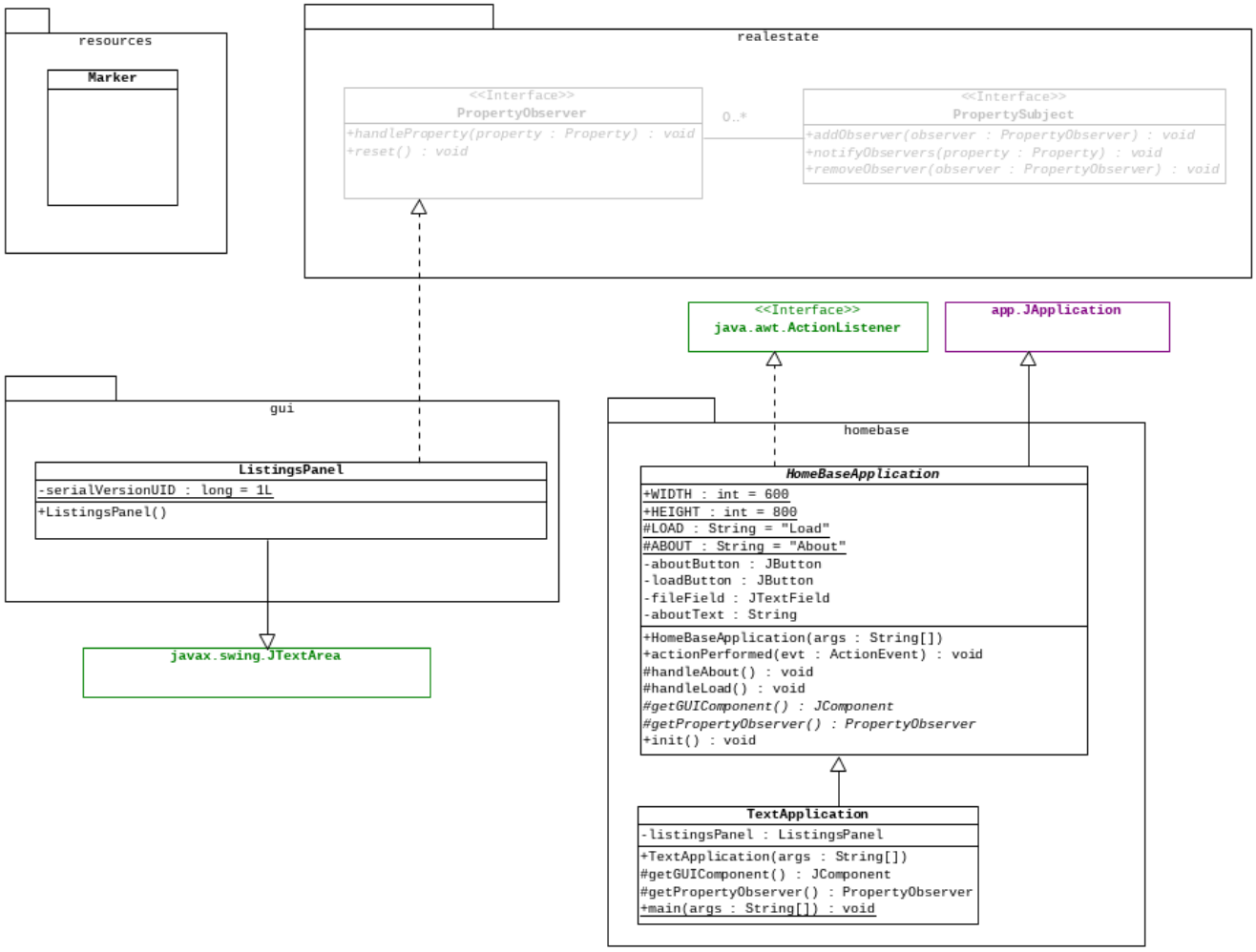
ID4. Clicking on the About button must cause a window that looks something like the following to be shown:



ID5. Clicking on the OK button in the About Window must cause it to close, returning control to the main window.

Class Diagram

The relationships between the various classes and interfaces of the system are illustrated in the following UML class diagram.



Classes/interfaces that are shown in jade green are part of the Java API. Classes/interfaces that are show in magenta are part of the Multimedia API. Existing classes/interface are show in gray. (Note: Many existing classes/interfaces have been omitted to simplify the diagram.)

In addition to the specifications that are contained in this class diagram, the implementation must comply with the following specifications.

The `ListingsPanel` Class

The `ListingsPanel` class is an encapsulation of the central GUI component.

Methods

```
    handleProperty(Property prop)
```

Must add a verbose `String` representation of the `Property` object to the component.

```
    reset()
```

Must clear all information that was being displayed in the component.

The `HomeBaseApplication` Class

The `HomeBaseApplication` class is the abstract parent of a variety of applications (including this one). It has the text field and buttons that allows the user to interact with the application.

Methods

```
    HomeBase(String[] args)
```

The explicit value constructor is passed the command-line arguments. It must pass them to its parent's constructor. It must then perform any necessary non-GUI-related initializations. (Remember that GUI-related initializations must be performed in the `init()` method.)

```
    actionPerformed()
```

Must handle the events generated by the user clicking on the About and/or Load buttons. When the user clicks on the About button, this method must invoke the `handleAbout()` method. When the user clicks on the Load button, this method must invoke the `handleLoad()` method.

```
    getGUIComponent()
```

The `getGUIComponent()` method is invoked when the application is being layed-out. In concrete specializations, this method returns the "main" `JComponent` (e.g., the `JComponent` that is used to display property information).

```
getPropertyObserver()
```

The `getPropertyObserver()` method is invoked when the application responds to the Load button. In concrete specializations, this method returns the `PropertyObserver` that becomes the observer of the `PropertySubject` (i.e., the `PropertyReader`).

```
handleAbout()
```

This method must invoke the `showMessageDialog()` method in the `JOptionPane` class, passing it the text contained in `about.txt`.

```
handleLoad()
```

This method must construct an appropriate kind of `PropertyReader` (based on the file type, either `.apartments` or `.houses`), reset the `PropertyObserver`, make the `PropertyObserver` an observer of the `PropertyReader`, and read the file.

```
init()
```

Must construct the GUI components and lay them out (as illustrated in the interaction design portion of this document).

The `TextApplication` Class

The `TextApplication` class is the "main" class of the application.

Methods

```
TextApplication(String[] args)
```

The explicit value constructor is passed the command-line arguments. It must pass them to its parent's constructor. It must then construct the `ListingsPanel` attribute.

```
getGUIComponent()
```

Must return the `ListingsPanel` attribute.

```
getPropertyObserver()
```

Must return the `ListingsPanel` attribute.

`main()`

In keeping with the application multimedia framework being used, this method must be implemented as follows:

```
public static void main(final String[] args)
{
  JApplication app = new TextApplication(args);
  invokeInEventDispatchThread(app);
}
```

Deployment

This application must be deployed as a single `.jar` file named `TextApplication.jar`. It must contain the file named `about.txt` in the resources package.

The two data files that were created for testing purposes must not be included in the .jar file (i.e., they must be read from the local file system).