Names and NSIDs: Wynston Ramsay - wcr723, Evan Snook - egs819, Darvin Zhang - ddz369
Project: jEdit
Change Number: 1
Date: 04/01/2018

Modify the splash window

1. Change Request:

Currently the splash window of jEdit is a static picture. Add the names and emails of your group members to it. And add moving text as the same effect shown in “About jEdit” dialog. Adjust the scrolling speed so that all text can be shown.

2. Concept Location:

The splash page was very intuitive to where it was located. By looking at the structure of the project, we were able to located the “gui” package. Lone and behold, the splash page was there. Other than that, it was also hinted that the AboutDialog had the animation that we needed to implement to the SplashScreen.java. It luckily also happened to be in the same package, saving us quite a bit of time as a result.

Table 1. The list of all the classes visited during concept location.

#	File name	Tool used	Located?	Comments
1	org.gjt.sp.jedit.gui/ AboutDialog.java	Eclipse Project Explorer	**Unchanged**	Assignment specification hinted that the splash screen changes were similar to the code in AbouDialog.java
2	org.gjt.sp.jedit.gui/ SplashScreen.java	Eclipse Project Explorer	**Changed**	Added all the required splash screen changes with the help of AboutDialog.java

3. Impact Analysis:

Table 2. The list of all the classes visited during impact analysis.

#	Class name	Tool used	Impacted?	Comments
1	org.gjt.sp.jedit.gui/ SplashScreen.java	Eclipse - CTRL + F	SplashScreen	This class contains all the necessary modifications for this change request

2	org.gjt.sp.jedit.gui/ AboutDialog.java	Eclipse Project Explorer	AboutDialog, AboutPanel, ActionHandler	Provides a good example of how to make an animation thread (useful for SplashScreen)

4. Learning process:

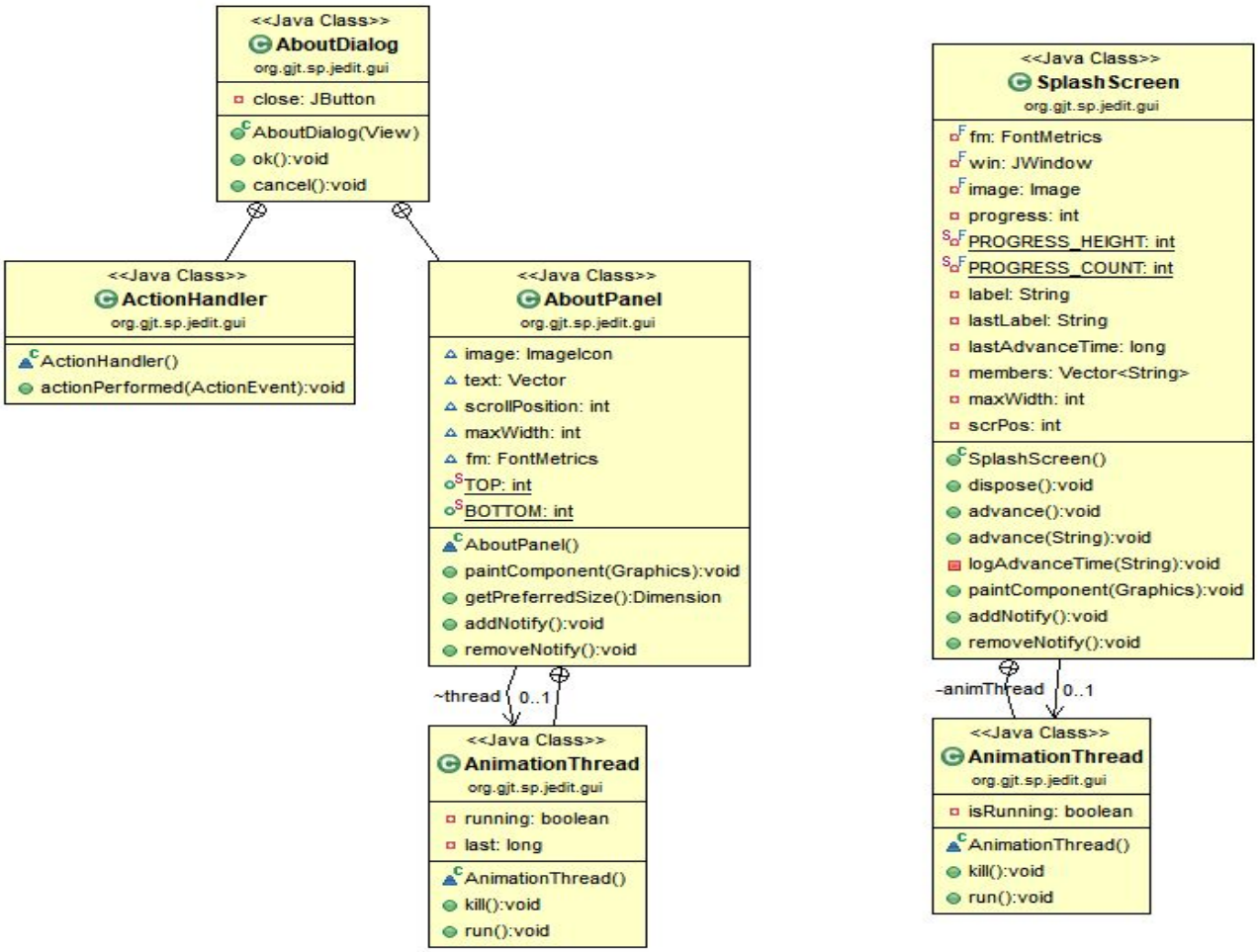

Figure 1: UML diagram of classes visited (generated by ObjectAid)

5. Description of the implementation:

All of our changes were made in SplashScreen.java. The assignment specification hint guided us to AboutDialog.java and gave us a good idea on how to replicate the features we needed. The first main change was making sure to add a vector containing our names and NSID's. Then we applied this vector to the splash screen with the use of an animation thread that was similar to AboutDialog.java's implementation.

6. Sources: jEdit, Eclipse, GitHub, ObjectAid UML Diagram (Eclipse Add-on)

7. Highlighted Source Code:

(Please note that there is quite a bit of automatic refactorization - generally, ignore red.)

All modifications were made in SplashScreen.java

(Please see the following pages for the GitHub generated diff file)

166 assignment 3/JEdit 4.3/org/gjt/sp/jedit/gui/SplashScreen.java

```
@@ -21,6 +21,7 @@
 import javax.swing.*;
 import java.awt.*;
+import java.util.Vector;

 import org.gjt.sp.jedit.jEdit;
 import org.gjt.sp.util.Log;
@@ -29,29 +30,37 @@
  * The splash screen displayed on startup.
  * @version $Id: SplashScreen.java 8122 2006-11-24 11:29:49Z kpouer $
  */
-public class SplashScreen extends JComponent
-{
-	public SplashScreen()
-	{
+public class SplashScreen extends JComponent {
+	public SplashScreen() {
 		setCursor(Cursor.getPredefinedCursor(Cursor.WAIT_CURSOR));
 		setBackground(Color.white);

-		Font font = new Font("Dialog",Font.PLAIN,10);
+		Font font = new Font("Dialog", Font.PLAIN, 10);
 		setFont(font);
 		fm = getFontMetrics(font);

-		image = getToolkit().getImage(
-			getClass().getResource("/org/gjt/sp/jedit/icons/splash.png"));
+		image = getToolkit().getImage(getClass().getResource("/org/gjt/sp/jedit/icons/splash.png"));
 		MediaTracker tracker = new MediaTracker(this);
-		tracker.addImage(image,0);
-
-		try
-		{
+		tracker.addImage(image, 0);
+
+		// #1 - create a vector to hold our group member names
+		members = new Vector(50);
+		members.addElement("Wynston Ramsay - wcr723");
+		members.addElement("Evan Snook - egs819");
+		members.addElement("Darvin Zhang - ddz369");
+
+		// #1 - initialize the starting position of the scrolling text
+		scrPos = -300;
```

```
54  +
55  +               // #1 - maxWidth, given the longest identification
56  +               maxWidth = fm.stringWidth("Wynston Ramsay - wcr723") + 10;
57  +
58  +               animThread = new AnimationThread();
59  +
60  +               try {
50  61                      tracker.waitForAll();
51      -               }
52      -               catch(Exception e)
53      -               {
54      -                       Log.log(Log.ERROR,this,e);
    62  +               } catch (Exception e) {
    63  +                       Log.log(Log.ERROR, this, e);
55  64              }
56  65
57  66              win = new JWindow();
    @@ -60,108 +69,140 @@ public SplashScreen()
60  69              int height = gs[0].getDisplayMode().getHeight();
61  70              int width = gs[0].getDisplayMode().getWidth();
62  71              Dimension screen = new Dimension(width, height);
63      -               Dimension size = new Dimension(image.getWidth(this) + 2,
64      -                       image.getHeight(this) + 2 + PROGRESS_HEIGHT);
    72  +               Dimension size = new Dimension(image.getWidth(this) + 2, image.getHeight(this) + 2 + PROGRESS_HEIGHT);
65  73              win.setSize(size);
66  74
67  75              win.getContentPane().add(this, BorderLayout.CENTER);
68  76
69      -               win.setLocation((screen.width - size.width) / 2,
70      -                       (screen.height - size.height) / 2);
    77  +               win.setLocation((screen.width - size.width) / 2, (screen.height - size.height) / 2);
71  78              win.validate();
72  79              win.setVisible(true);
73  80      }
74  81
75      -       public void dispose()
76      -       {
    82  +       public void dispose() {
77  83              win.dispose();
78  84      }
79  85
80      -       public synchronized void advance()
81      -       {
    86  +       public synchronized void advance() {
82  87              logAdvanceTime(null);
83  88              progress++;
```

```
78  84          }
79  85
80      -       public synchronized void advance()
81      -       {
        86  +       public synchronized void advance() {
82  87                  logAdvanceTime(null);
83  88                  progress++;
84  89                  repaint();
85  90
86  91                  // wait for it to be painted to ensure progress is updated
87  92                  // continuously
88      -               try
89      -               {
        93  +               try {
90  94                          wait();
91      -               }
92      -               catch(InterruptedException ie)
93      -               {
94      -                       Log.log(Log.ERROR,this,ie);
        95  +               } catch (InterruptedException ie) {
        96  +                       Log.log(Log.ERROR, this, ie);
95  97                  }
96  98          }
97  99
98      -       public synchronized void advance(String label)
99      -       {
        100 +       public synchronized void advance(String label) {
100 101                 logAdvanceTime(label);
101 102                 progress++;
102 103                 this.label = label;
103 104                 repaint();
104 105
105 106                 // wait for it to be painted to ensure progress is updated
106 107                 // continuously
107     -               try
108     -               {
        108 +           try {
109 109                         wait();
110     -               }
111     -               catch(InterruptedException ie)
112     -               {
113     -                       Log.log(Log.ERROR,this,ie);
        110 +           } catch (InterruptedException ie) {
        111 +                   Log.log(Log.ERROR, this, ie);
114 112                 }
115 113         }
116 114
```

```
117     -       private void logAdvanceTime(String label)
118     -       {
        115 +   private void logAdvanceTime(String label) {
119     116             long currentTime = System.currentTimeMillis();
120     -               if (lastLabel != null)
121     -               {
122     -                       Log.log(Log.DEBUG, SplashScreen.class, lastLabel +':'+(currentTime - lastAdvanceTime) + "ms");
        117 +           if (lastLabel != null) {
        118 +                   Log.log(Log.DEBUG, SplashScreen.class, lastLabel + ':' + (currentTime - lastAdvanceTime) + "ms");
123     119             }
124     -               if (label != null)
125     -               {
        120 +           if (label != null) {
126     121                     lastLabel = label;
127     122                     lastAdvanceTime = currentTime;
128     123
129     124             }
130     125
131     126     }
132     127
133     -       public synchronized void paintComponent(Graphics g)
134     -       {
        128 +   public synchronized void paintComponent(Graphics g) {
135     129             Dimension size = getSize();
136     130
137     131             g.setColor(Color.black);
138     -               g.drawRect(0,0,size.width - 1,size.height - 1);
        132 +           g.drawRect(0, 0, size.width - 1, size.height - 1);
139     133
140     -               g.drawImage(image,1,1,this);
        134 +           g.drawImage(image, 1, 1, this);
141     135
142     136             // XXX: This should not be hardcoded
143     137             g.setColor(Color.white);
144     -               g.fillRect(1,image.getHeight(this) + 1,
145     -                       ((win.getWidth() - 2) * progress) / PROGRESS_COUNT, PROGRESS_HEIGHT);
        138 +           g.fillRect(1, image.getHeight(this) + 1, ((win.getWidth() - 2) * progress) / PROGRESS_COUNT, PROGRESS_HEIGHT);
146     139
147     140             g.setColor(Color.black);
148     141
149     -               if (label != null)
150     -               {
151     -                       g.drawString(label,
152     -                                       (getWidth() - fm.stringWidth(label)) / 2,
153     -                                       image.getHeight(this) + (PROGRESS_HEIGHT
154     -                                                       + fm.getAscent() + fm.getDescent()) / 2);
```

```
142 +               if (label != null) {
143 +                       g.drawString(label, (getWidth() - fm.stringWidth(label)) / 2,
144 +                                       image.getHeight(this) + (PROGRESS_HEIGHT + fm.getAscent() + fm.getDescent()) / 2);
155 145             }
156 146
157     -
158 147             String version = jEdit.getVersion();
159     -           g.drawString(version,
160     -                   getWidth() - fm.stringWidth(version) - 2,
161     -                   image.getHeight(this) - fm.getDescent());
148 +           g.drawString(version, getWidth() - fm.stringWidth(version) - 2, image.getHeight(this) - fm.getDescent());
149 +
150 +           // #1 - Draw the our members out, per line, given a starting position
151 +           int height = fm.getHeight();
152 +           int firstLine = scrPos / height;
153 +           int lines = getHeight() / height;
154 +           int y = fm.getHeight();
155 +           for (int i = 0; i <= lines; i++) {
156 +                   if (i + firstLine >= 0 && i + firstLine < members.size()) {
157 +                           String line = (String)members.get(i + firstLine);
158 +                           g.drawString(line, 20, y);
159 +                   }
160 +                   y += fm.getHeight();
161 +           }
162 +
162 163             notify();
163 164     }
164 165
166 +   public void addNotify() {
167 +           super.addNotify();
168 +           animThread.start();
169 +   }
170 +
171 +   public void removeNotify() {
172 +           super.removeNotify();
173 +           animThread.kill();
174 +   }
175 +
176 +   // #1 - AnimationThread. Continually move the screen
177 +   // position of our lines upwards, and redraw them
178 +   // to create a scrolling effect
179 +   public class AnimationThread extends Thread {
180 +           private boolean isRunning;
181 +
182 +           AnimationThread() {
183 +                   isRunning = true;
```

```
184 +                       setPriority(Thread.MIN_PRIORITY);
185 +               }
186 +
187 +               public void kill() {
188 +                       isRunning = false;
189 +               }
190 +
191 +               public void run() {
192 +                       while (isRunning) {
193 +                               scrPos += 5;
194 +
195 +                               try {
196 +                                       animThread.sleep(10);
197 +                               } catch (Exception ex) {
198 +                                       System.out.println(ex.toString());
199 +                               }
200 +
201 +                               repaint(getWidth() / 2 - maxWidth, 0, maxWidth * 2, getHeight());
202 +                       }
203 +               }
204 +       }
205 +
165 206         // private members
166 207         private final FontMetrics fm;
167 208         private final JWindow win;
    @@ -172,4 +213,9 @@ public synchronized void paintComponent(Graphics g)
172 213         private String label;
173 214         private String lastLabel;
174 215         private long lastAdvanceTime = System.currentTimeMillis();
216 +       // #1 - variables
217 +       private Vector<String> members;
218 +       private int maxWidth;
219 +       private int scrPos;
220 +       private AnimationThread animThread;
175 221  }
```