jEdit4.3 Feature Request: Count Words per Minute

Peter Johnson

August 16, 2013

I implemented all four features. The core of the feature is a new class, WpmCounter. WpmCounter keeps track of the time that the last input was receieved to determine when to stop updating, and also the time that it started updating to aid in the calculations. Inside a thread, WpmCounter repeatedly checks if it should stop updating, and otherwise calculates the wpm and updates the status bar.

My rationale for the WpmCounter class was to encapsualate all of the WPM calculations in one place, and to have a thread so that the WPM can be regularly updated independently from the rest of the system. WpmCounter is intended solely for calculating the WPM, and the code that does the actual updating of the status bar is in the StatusBar instead, where it belongs.

One limitation is that it calculates the WPM based on the cumulative time since it began updating, so if elapsedTimeBeforeStopUpdating is too high, the WPM might be inaccurate because it fails to reflect the time that you weren't typing.

Another possible limitation is that it registers words as the first space character after one or more non-space characters, so your WPM is dependent on the length of the words you are typing. This is of course the intuitive definition of WPM, but I have seen other WPM counters that define a specific length of characters to represent a word to make WPM a more standardized measurment.

A third limitation is that the WPM will not actually stop updating until the next time that the WPM would have been drawn to the status bar. This is not an issue for reasonable values, but if the refresh rate is significantly longer than elapsedTimeBeforeStopUpdating then the WPM display will not go away when expected. This could be corrected by sleeping for shorter intervals to check whether to stop more frequently.

Here are the diffs of my code: I apologize that some of the longer lines are cut off.

```
diff --git a/jEdit/org/gjt/sp/jedit/View.java b/jEdit/org/gjt/sp/jedit/View.java
index 650e03b..bc68fcc 100644
--- a/jEdit/org/gjt/sp/jedit/View.java
+++ b/jEdit/org/gjt/sp/jedit/View.java
@@ -444,6 +444,18 @@ public class View extends JFrame implements EBComponent, InputHandlerProv
  return status;
  } //}}}

+    /* pajohnson@email.wm.edu
+     * A simple getter method to get the wpm counter.
+     */
+    //{{{ getWpmCounter() method
+    /**
```

```
+     * Returns the WPM counter.
+     */
+    public WpmCounter getWpmCounter()
+    {
+        return wpmCounter;
+    } //}}}
+
  //{{{ quickIncrementalSearch() method
  /**
   * Quick search.
@@ -1348,6 +1360,10 @@ public class View extends JFrame implements EBComponent, InputHandlerPr
  toolBarManager = new ToolBarManager(topToolBars, bottomToolBars);

  status = new StatusBar(this);
+        /* pajohnson@email.wm.edu
+         * Creating our WpmCounter instance with the status bar.
+         */
+        wpmCounter = new WpmCounter(status);

  inputHandler = new DefaultInputHandler(this,(DefaultInputHandler)
  jEdit.getInputHandler());
@@ -1556,6 +1572,10 @@ public class View extends JFrame implements EBComponent, InputHandlerPr
  private final BufferSet localBufferSet;

  private StatusBar status;
+    /* pajohnson@email.wm.edu
+     * Declaring wpmCounter.
+     */
+    private WpmCounter wpmCounter;

  private InputHandler inputHandler;
  private Macros.Recorder recorder;
diff --git a/jEdit/org/gjt/sp/jedit/WpmCounter.java b/jEdit/org/gjt/sp/jedit/WpmCounter.java
new file mode 100644
index 0000000..2a160ef
--- /dev/null
+++ b/jEdit/org/gjt/sp/jedit/WpmCounter.java
@@ -0,0 +1,102 @@
+/* pajohnson@email.wm.edu
+ * This file is to encapsulate all the wpm counting functionality.
+ */
+package org.gjt.sp.jedit;
+
+import java.util.Date;
+import org.gjt.sp.jedit.gui.StatusBar;
+
+public class WpmCounter implements Runnable
+{
+
+    public WpmCounter(StatusBar status)
```

```
+    {
+        this.status = status;
+        lastActivity = 0;
+        Thread thread = new Thread(this);
+        thread.setDaemon(true);
+        thread.start();
+    }
+
+    public void setStopDelay(int stopDelay)
+    {
+        this.stopDelay = stopDelay;
+    }
+
+    public void setRefreshRate(int refreshRate)
+    {
+        this.refreshRate = refreshRate;
+    }
+
+    // Tell the counter to count this char.
+    public void countChar(char ch)
+    {
+        charactersCounted += 1;
+
+        boolean isSpace = Character.isWhitespace(ch);
+        // A space usually indicates a new word was finished being typed,
+        // but not if several spaces come in a row.
+        if (isSpace && !lastCharWasSpace)
+            wordsCounted += 1;
+
+        lastCharWasSpace = isSpace;
+
+        long now = new Date().getTime();
+
+        // If going from inactive to active, save the start time
+        // and reset the counts.
+        if (!active)
+        {
+            startTime = now;
+            wordsCounted = 0;
+            charactersCounted = 0;
+        }
+
+        active = true;
+        lastActivity = now;
+    }
+
+    public void run()
+    {
+        while (true)
+        {
```

```
+            if (active) {
+                long now = new Date().getTime();
+
+                // Clear the status because we haven't been typing in a while.
+                if (now - lastActivity > stopDelay)
+                {
+                    status.updateWpmStatus();
+                    active = false;
+                }
+                else
+                {
+                    // Calculate the wpm and cpm.
+                    double factor = (now - startTime) / 60000.0;
+                    int wpm = (int)(wordsCounted / factor);
+                    int cpm = (int)(charactersCounted / factor);
+
+                    // Update the status bar and clear the count.
+                    status.updateWpmStatus(wpm, cpm);
+                }
+            }
+
+            try
+            {
+                Thread.sleep(refreshRate);
+            } catch (InterruptedException e) {}
+        }
+    }
+
+    // Private instance fields.
+    private StatusBar status;
+    private volatile long startTime;
+    private volatile long lastActivity;
+    private volatile boolean active;
+    private volatile int wordsCounted;
+    private volatile int charactersCounted;
+    private boolean lastCharWasSpace;
+    private int stopDelay;
+    private int refreshRate;
+
+}
diff --git a/jEdit/org/gjt/sp/jedit/gui/StatusBar.java b/jEdit/org/gjt/sp/jedit/gui/StatusBar.
index 0bb6f97..fd9c0a6 100644
--- a/jEdit/org/gjt/sp/jedit/gui/StatusBar.java
+++ b/jEdit/org/gjt/sp/jedit/gui/StatusBar.java
@@ -40,12 +40,16 @@ import org.gjt.sp.jedit.gui.statusbar.ToolTipLabel;
 import org.gjt.sp.util.*;
 //}}}

+/* pajohnson@email.wm.edu
+ * Added that the status bar displays the WPM and CPM to the javadoc.
```

```
+ */
 /**
  * The status bar used to display various information to the user.<p>
  *
  * Currently, it is used for the following:
  * <ul>
  * <li>Displaying caret position information
+ * <li>Displaying words per minute and characters per minute
  * <li>Displaying {@link InputHandler#readNextChar(String,String)} prompts
  * <li>Displaying {@link #setMessage(String)} messages
  * <li>Displaying I/O progress
@@ -77,11 +81,21 @@ public class StatusBar extends JPanel implements WorkThreadProgressListene

  MouseHandler mouseHandler = new MouseHandler();

+        statusBox = new Box(BoxLayout.X_AXIS);
+        panel.add(BorderLayout.WEST, statusBox);
+
  caretStatus = new ToolTipLabel();
  caretStatus.setName("caretStatus");
  caretStatus.setToolTipText(jEdit.getProperty("view.status.caret-tooltip"));
  caretStatus.addMouseListener(mouseHandler);

+        /* pajohnson@email.wm.edu
+         * Add a tooltip to explain WPM and CPM info.
+         */
+        wpmStatus = new ToolTipLabel();
+        wpmStatus.setName("wpmStatus");
+        wpmStatus.setToolTipText(jEdit.getProperty("view.status.wpm-tooltop"));
+
  message = new JLabel(" ");
  setMessageComponent(message);

@@ -95,6 +109,10 @@ public class StatusBar extends JPanel implements WorkThreadProgressListene
  lineSepWidget = _getWidget("lineSep");
  } //}}}

+    /* pajohnson@email.wm.edu
+     * I've changed the construction of the status bar to having
+     * a box to the WEST containing the caret and wpm status tooltips.
+     */
  //{{{ propertiesChanged() method
  public void propertiesChanged()
  {
@@ -102,11 +120,26 @@ public class StatusBar extends JPanel implements WorkThreadProgressListe
  Color bg = jEdit.getColorProperty("view.status.background");

  showCaretStatus = jEdit.getBooleanProperty("view.status.show-caret-status");
+        /* pajohnson@email.wm.edu
+         * Get the new values of the various wpm options, and update the
```

```
+         * stop delay and refresh rate with the counter itself.
+         */
+ showWpm = jEdit.getBooleanProperty("view.status.show-wpm");
+ showCpm = jEdit.getBooleanProperty("view.status.show-cpm");
+        WpmCounter wpmCounter = view.getWpmCounter();
+        wpmCounter.setStopDelay(Integer.parseInt(jEdit.getProperty("view.status.wpm-stop-dela
+        wpmCounter.setRefreshRate(Integer.parseInt(jEdit.getProperty("view.status.wpm-refresh

  panel.setBackground(bg);
  panel.setForeground(fg);
  caretStatus.setBackground(bg);
  caretStatus.setForeground(fg);
+        /* pajohnson@email.wm.edu
+         * Set the background and foreground of wpmStatus, just like they
+         * did for caretStatus.
+         */
+        wpmStatus.setBackground(bg);
+        wpmStatus.setForeground(fg);
  message.setBackground(bg);
  message.setForeground(fg);

@@ -115,9 +148,14 @@ public class StatusBar extends JPanel implements WorkThreadProgressListen
  //UIManager.getFont("Label.font");
  FontMetrics fm = getFontMetrics(font);

+        /* pajohnson@email.wm.edu
+         * Emptying the status box before adding them back.
+         */
+        statusBox.removeAll();
+
  if (showCaretStatus)
  {
-panel.add(BorderLayout.WEST,caretStatus);
+ statusBox.add(caretStatus);

  caretStatus.setFont(font);

@@ -126,8 +164,22 @@ public class StatusBar extends JPanel implements WorkThreadProgressListen
  caretStatus.setPreferredSize(dim);
  updateCaretStatus();
  }
-else
-panel.remove(caretStatus);
+
+        statusBox.add(Box.createGlue());
+
+        /* pajohnson@email.wm.edu
+         * Add the wpm tooltop object to the status bar.
+         */
+        if (showWpm || showCpm)
```

```
+        {
+            statusBox.add(wpmStatus);
+
+            wpmStatus.setFont(font);
+
+            Dimension dim = new Dimension(fm.stringWidth(wpmTestStr),
+                    fm.getHeight());
+            wpmStatus.setPreferredSize(dim);
+        }

  String statusBar = jEdit.getProperty("view.status");
  if (!StandardUtilities.objectsEqual(currentBar, statusBar))
@@ -384,6 +436,40 @@ public class StatusBar extends JPanel implements WorkThreadProgressListen
   }
  } //}}}

+    /* pajohnson@email.wm.edu
+     * This function is responsible for updating the wpm display on
+     * the status bar. It is passed the wpm and cpm as arguments,
+     * and then should decide whether and how to display them. The no
+     * arguments verion removes the display.
+     */
+ //{{{ updateWpmStatus() method
+    public void updateWpmStatus() {
+        wpmStatus.setText(" ");
+    }
+
+    public void updateWpmStatus(int wpm, int cpm)
+    {
+        if ((showWpm && wpm >= 0) || (showCpm && cpm >= 0)) {
+
+            if (showWpm) {
+                buf.append("WPM: ");
+                buf.append(Integer.toString(wpm));
+                if (showCpm) {
+                    // Append separator if both are being shown.
+                    buf.append(" | ");
+                }
+            }
+
+            if (showCpm) {
+                buf.append("CPM: ");
+                buf.append(Integer.toString(cpm));
+            }
+
+            wpmStatus.setText(buf.toString());
+            buf.setLength(0);
+        }
+    } //}}}
+
```

```
  //{{{ updateBufferStatus() method
  public void updateBufferStatus()
  {
@@ -407,7 +493,12 @@ public class StatusBar extends JPanel implements WorkThreadProgressListen
  private final View view;
  private final JPanel panel;
  private final Box box;
+    /* pajohnson@email.wm.edu
+     * Declaring wpm tooltop, and box for both tooltips.
+     */
+    private final Box statusBox;
  private final ToolTipLabel caretStatus;
+ private final ToolTipLabel wpmStatus;
  private Component messageComp;
  private final JLabel message;
  private final Widget modeWidget;
@@ -425,10 +516,20 @@ public class StatusBar extends JPanel implements WorkThreadProgressListe
  private final Segment seg = new Segment();

  private boolean showCaretStatus;
+    /* pajohnson@email.wm.edu
+     * Flags for whether or not to display WPM and CPM.
+     */
+    private boolean showWpm;
+    private boolean showCpm;
  //}}}

  //static final String caretTestStr = "99999999,9999,999-999 99%";
  static final String caretTestStr = "9999,999-999 (99999999/99999999)";
+    /* pajohnson@email.wm.edu
+     * This string is for approximating the maximum size that the wpm
+     * info will take up on the status bar.
+     */
+    static final String wpmTestStr = "WPM: 9999 | CPM: 9999";

  //{{{ getWidget() method
  private Widget getWidget(String name)
diff --git a/jEdit/org/gjt/sp/jedit/jedit_gui.props b/jEdit/org/gjt/sp/jedit/jedit_gui.props
index 4498472..791f7bb 100644
--- a/jEdit/org/gjt/sp/jedit/jedit_gui.props
+++ b/jEdit/org/gjt/sp/jedit/jedit_gui.props
@@ -2066,6 +2066,20 @@ options.status.caret.dot=Show caret offset from start of line
 options.status.caret.virtual=Show caret virtual offset from start of line
 options.status.caret.offset=Show caret offset from start of file
 options.status.caret.bufferlength=Show length of file
+
+# pajohnson@email.wm.edu
+# I'm defining here default values for all new gui components, so that
+# the menu can be easily updated later by modifying these values.
+view.status.wpm-tooltip=WPM: <words per minute> | CPM: <characters per minute>
```

```
+options.status.wpm.title=WPM options:
+options.status.wpm.showWpm=Show number of words per minute (WPM)
+view.status.show-wpm=false
+options.status.wpm.showCpm=Show number of characters per minute (CPM)
+view.status.show-cpm=false
+options.status.wpm.stopDelay=Elapsed miliseconds before stopping updating WPM and CPM
+view.status.wpm-stop-delay=2000
+options.status.wpm.refreshRate=Refresh rate (Miliseconds)
+view.status.wpm-refresh-rate=1000
 #}}}

 #{{{ Syntax Highlighting pane
diff --git a/jEdit/org/gjt/sp/jedit/options/StatusBarOptionPane.java b/jEdit/org/gjt/sp/jedit/
index 5d4bdcf..bce5ceb 100644
--- a/jEdit/org/gjt/sp/jedit/options/StatusBarOptionPane.java
+++ b/jEdit/org/gjt/sp/jedit/options/StatusBarOptionPane.java
@@ -126,6 +126,30 @@ public class StatusBarOptionPane extends AbstractOptionPane
  optionsPanel.addComponent(showCaretOffset);
  optionsPanel.addComponent(showCaretBufferLength);

+        /* pajohnson@email.wm.edu
+         * I'm instantiating the various gui components here, opting for
+         * default values from properties if available. I also add them to
+         * the panel. This is all done in a way similar to how other gui
+         * components were added.
+         */
+ optionsPanel.addSeparator();
+ optionsPanel.addComponent(new JLabel(jEdit.getProperty("options.status.wpm.title", "WPM opti
+ showWpm = new JCheckBox(jEdit.getProperty("options.status.wpm.showWpm", "Show number of word
+ jEdit.getBooleanProperty("view.status.show-wpm", true));
+        showWpm.setName("showWpm");
+        optionsPanel.addComponent(showWpm);
+ showCpm = new JCheckBox(jEdit.getProperty("options.status.wpm.showCpm", "Show number of char
+ jEdit.getBooleanProperty("view.status.show-cpm", true));
+        showCpm.setName("showCpm");
+        optionsPanel.addComponent(showCpm);
+        optionsPanel.addComponent(new JLabel(jEdit.getProperty("options.status.wpm.stopDelay"
+        wpmStopDelay = new JTextField(jEdit.getProperty("view.status.wpm-stop-delay", "2000")
+        wpmStopDelay.setName("wpmStopDelay");
+        optionsPanel.addComponent(wpmStopDelay);
+        optionsPanel.addComponent(new JLabel(jEdit.getProperty("options.status.wpm.refreshRat
+        wpmRefreshRate= new JTextField(jEdit.getProperty("view.status.wpm-refresh-rate", "100
+        wpmRefreshRate.setName("wpmRefreshRate");
+        optionsPanel.addComponent(wpmRefreshRate);
  //}}}


@@ -227,6 +251,13 @@ public class StatusBarOptionPane extends AbstractOptionPane
  jEdit.setBooleanProperty("view.status.show-caret-offset", showCaretOffset.isSelected());
  jEdit.setBooleanProperty("view.status.show-caret-bufferlength", showCaretBufferLength.isSele
```

```
+        /* pajohnson@email.wm.edu
+         * Writing the code to save our new wpm values.
+         */
+        jEdit.setBooleanProperty("view.status.show-wpm", showWpm.isSelected());
+        jEdit.setBooleanProperty("view.status.show-cpm", showCpm.isSelected());
+        jEdit.setProperty("view.status.wpm-stop-delay", wpmStopDelay.getText());
+        jEdit.setProperty("view.status.wpm-refresh-rate", wpmRefreshRate.getText());
  } //}}}

  //{{{ Private members
@@ -250,6 +281,14 @@ public class StatusBarOptionPane extends AbstractOptionPane
  private JCheckBox showCaretVirtual;
  private JCheckBox showCaretOffset;
  private JCheckBox showCaretBufferLength;
+
+    /* pajohnson@email.wm.edu
+     * Declaring the gui components we're going to use.
+     */
+    private JCheckBox showWpm;
+    private JCheckBox showCpm;
+    private JTextField wpmStopDelay;
+    private JTextField wpmRefreshRate;
  //}}}

  //{{{ updateButtons() method
diff --git a/jEdit/org/gjt/sp/jedit/textarea/JEditTextArea.java b/jEdit/org/gjt/sp/jedit/texta
index c6ff6b4..ca6baa6 100644
--- a/jEdit/org/gjt/sp/jedit/textarea/JEditTextArea.java
+++ b/jEdit/org/gjt/sp/jedit/textarea/JEditTextArea.java
@@ -217,11 +217,15 @@ public class JEditTextArea extends TextArea
  }
  } //}}}

+    /* pajohnson@email.wm.edu
+     * Count the character for WPM, and update the javadoc.
+     */
  //{{{ userInput() method
  /**
   * Handles the insertion of the specified character. It performs the
   * following operations in addition to TextArea#userInput(char):
   * <ul>
+     * <li>Count the character for determining WPM
   * <li>Inserting a space with automatic abbrev expansion enabled will
   * try to expand the abbrev
   * </ul>
@@ -236,6 +240,8 @@ public class JEditTextArea extends TextArea
  && Abbrevs.expandAbbrev(view,false))
  return;
```

```
+         view.getWpmCounter().countChar(ch);
+
  super.userInput(ch);
  } //}}}
```