```java
package rolodex;

import javax.swing.*;
import java.awt.event.*;

/****
 *
 * Class CommandEdTextArea is a extension of JTextArea that specializes the
 * processComponentKeyEvent method to look for input of a newline character.
 * When the newline is encountered, the command line is parsed and the
 * appropriate Rolodex model function is called if the parse succeeds.
 *
 */

public class CommandEdTextArea extends JTextArea {

    /**
     * Construct this with the given rows, columns, parent view,
     * greatgrandparent view, and companion process.  The rows and columns are
     * set by passing the buck to the parent constructor.  The parent view is
     * for getting the command line text.  The greatgrandparent view is for
     * error message reporting.  The companion process is for parsing the
     * command line text and calling the model if the parse succeeds.
     */
    CommandEdTextArea(int rows, int columns,
            RolodexCommandEditor rce, RolodexMultiViewMenuUI rmui,
                RolodexCommandProcessor rcp) {

        super(rows, columns);
        this.rce = rce;
        this.rmui = rmui;
        this.rcp = rcp;

    }

    /**
     * Examine each keyboard event, looking for a newline character (i.e.,
     * KeyEvent.VK_ENTER).  When one is detected, get the line typed in by the
     * user and send it to the text command processor which will in turn call
     * the appropriate model function if the command is valid.  Note that this
     * method is called three times for each user-invoked key action -- once
     * for pressed, once for released, and once for typed.  Only one of these
     * (typed) is responded to here, the other two ignored.
     */
    public void processComponentKeyEvent(KeyEvent e) {

        /*
         * Unconditionally call the parent version of this method to have the
         * key stroke properly dealt with, which typically means inserting the
         * character into the window.
         */
        super.processComponentKeyEvent(e);

        /*
         * Respond only to the "typed" event, thereby ignoring the "pressed"
         * and "released" events.  Per the specs of class KeyEvent, the "typed"
         * event is recognized by a value of VK_UNDEFINED returned from
         * KeyEvent.getKeyCode().
         */
        if (e.getKeyCode() == e.VK_UNDEFINED) {

            /*
             * Get the character associated with this event, i.e., the
             * character typed on the keyboard.
             */
            char c = e.getKeyChar();

            /*
             * Respond if the input character is ENTER.
             */
            if (c == e.VK_ENTER) {

                /*
                 * Call the command processor to parse and execute the command
                 * line.  Catch any input errors, if thrown, and ask the
                 * grandparent view to display them.
                 */
                try {
                    rcp.processCommand(rce.getCommandLine());
                }
                catch (InputErrors ie) {
                    /*
                     * Display zero or more error messages, as appropriate.
                     */
                    rmui.showErrorDialog(ie.getErrorList());
                }
            }
        }
    }

    /** The grandparent view. */
    protected RolodexCommandEditor rce;

    /** The grandparent view. */
    protected RolodexMultiViewMenuUI rmui;

    /** Companion process class. */
    RolodexCommandProcessor rcp;

}
```