Homework #G Lexi Interactive Input

Issued: Tuesday, April 2
Due: Tuesday, April 23

Purpose

This assignment allows you to extend Lexi to accept keyboard input. You will find (at least) these design patterns useful: Prototype(117), Command(233), and Singleton(127).

Assignment

Design (in UML) and implement (in Java) features that allow a Lexi user to:

- Select a position in the document, by clicking the mouse on a character.
- Insert characters in the document, at that position, by pressing keys on the keyboard. Characters appear as they are typed.

Let's call the selected position the *insertion point*.

Normal keystrokes, between " " (space) and "~" (twiddle) in the usual ASCII collating sequence, are simply inserted into the document at the insertion point. Certain "control characters" perform the operations shown in the table below. Other characters are ignored.

`^b`	move backward
`^f`	move forward
`^h`	remove (backspace/delete)
`^z`	insert new rectangle
`^x`	insert new row
`^c`	insert new column
`^v`	insert new scroller

A character can be converted to its corresponding control character with this method:

```
private static char ctl(char c) {
    return (char)(c&'\u001f');
}
```

Notes and Suggestions

- Think about what state needs to be maintained to represent the insertion point.
- Where should this state be stored?
- As with our buttons, a mouse click should execute a command.
- Likewise, a keystroke should execute a command.
- You may find that using a special character to mark the end of a row/column simplifies implementation.