Homework #1 Lexi, Composite, and Strategy

Issued: Tuesday, February 6
Due: Thursday, February 22

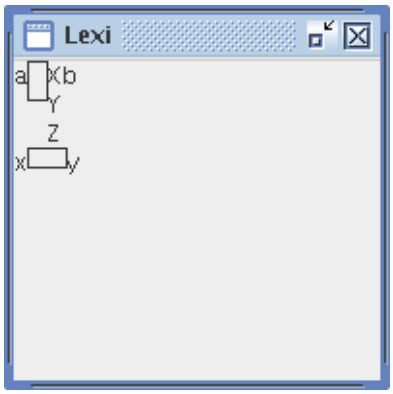
Purpose

This assignment allows you to learn about two design patterns: Composite(163) and Strategy(315).

Assignment

Design (in UML) and implement (in Java) the Document Structure and Formatting parts of the Lexi editor, as described in Sections 2.2 and 2.3 of our textbook.

Test your solution with a simple graphical demonstration. For example, this is the result of my test:



This document is a column containing two rows. The first row contains: a character (a), a rectangle, a column of three characters (X, Y, and Z), and another character (b).

Other Requirements

Pay particular attention to the intent of Strategy(315), as specialized for Lexi on page 42:

> *The Compositor-Composition class split ensures a strong separation between code that supports the document's physical structure and the code for different formatting algorithms. We can add new Compositor subclasses without touching the Glyph classes, and vice versa.*

Take this as friendly advice: Failure to design accordingly is the biggest single mistake you can make, in this course.

Also:

- Use the design suggestions from our textbook.
- Use the `Window` interface and `SwingWindow` implementation in:

  `pub/hw1`

  The `SwingWindow` constructor takes a window-name argument. You can remove the packaging stuff, at the top, if you wish.

- Since `Window` has a `setContents()` method, you should not call `draw()` on your top-level `Glyph`. `SwingWindow` does that for you, whenever redrawing is necessary.
- Design and implement the following subclasses of `Glyph`: `Character`, `Rectangle`, `Row`, and `Column`.
- If a client tries to perform an operation on an object of an inappropriate subclass of `Glyph`, throw an exception, as suggested on page 169, paragraph 2.
- Use Java generics, as appropriate.
- Casting is prohibited!
- Reflection (e.g., `instanceof`) is prohibited!
- Implement a *very* simple `Compositor`:
  - Do *not* add `Row` and `Column` objects, as suggested in the textbook.
  - Iterate through the `Composition`'s children, computing widths and heights, and adjusting positions accordingly.
  - Set the `Composition`'s width and height.
  - Process the `Composition`'s parent.