Übungsaufgabe #3: Whiteboard - Laden/Speichern *0 Punkte* 29.4.2002

In dieser Aufgabe soll das Whiteboard aus der vorherigen Aufgabe erweitert werden. Kopiere dazu alle Dateien aus Aufgabe 2 in das Verzeichnis `aufgabe3`. Es kann auch die Musterlösung verwendet werden, die unter `/proj/i4oovs/pub/aufgabe2` (ab 4.5.2002) verfügbar ist.

a) Erweitere die Klasse `Shape` um eine neue Methode `void resize(int w, int h)`. Neben der Position soll nun auch die Größe eines Grafikobjekts, wie folgt, auf der Kommandozeile übergeben werden können (Klassenname X Y Breite Höhe):
`java WhiteBoard Rect 120 80 20 30 Circle 50 150 60 50`
Die `resize`-Methode soll dazu verwendet werden um die Größe eines Grafikobjekts zu verändern. (Die Tatsache, dass Kreise nun auch oval sein können soll dabei ignoriert werden.) Fehler, die während der Erzeugung der verschiedenen Objekte auftreten, sollen zu einer entsprechenden Fehlermeldung führen und das Programm beenden.

b) Nun soll die Anwendung um eine grafische Oberfläche erweitert werden. Im Verzeichnis `/proj/i4oovs/pub/aufgabe3` befinden sich einige Dateien, welche die Gestaltung der Oberfläche übernehmen. Die Verwaltung der Shape-Objekte soll nun in ein Objekt des Typs `ShapeContainer` ausgelagert werden. Dafür soll die Klasse `WhiteBoard` aus Teilaufgabe a) in `ShapeContainerImpl` umbenannt werden. Außerdem soll sie in Teilaufgabe c) so angepasst werden, dass sie konform zum Interface `ShapeContainer` ist.
Vorab ist eine Test-Klasse `ShapeContainerImplTest` zu erstellen, welche folgende Methoden von `ShapeContainer` überprüft:

- `public void addShape(String spec) throws FormatException;`
  wird aufgerufen um ein neues Grafikobjekt aufgrund eines Klassennamens zu erzeugen. Der `spec`-String ist im gleichen Format wie bei der Spezifikation eines Grafikobjektes auf der Kommandozeile (Klassenname X Y Breite Höhe). Falls während der Erzeugung der Objekte oder der Auswertung der Koordinaten ein Fehler auftritt, soll eine `FormatException` geworfen werden. Die Klasse `FormatException` muss implementiert werden.
- `public Shape getShape(int x, int y);`
  wird verwendet um ein Grafikobjekt an den Koordinaten (x,y) zu selektieren
- `public void deleteShapes(int x, int y);`
  wir aufgerufen um alle Objekte die den Punktes (x,y) enthalten zu löschen

c) Die in Teilaufgabe b) genannten Methoden des Interface `ShapeContainer` sollen nun von der Klasse `ShapeContainerImpl` implementiert werden. Außerdem muss folgende Methode zum zeichnen der Grafikobjekte bereitgestellt werden:

- `public void paintDrawing(Graphics g);`
  wird von der `paint` -Methode aufgerufen um alle Grafikobjekte zu zeichnen

Alle weiteren Methoden des Interface können vorerst als leere Methoden implementiert werden.

d) Die Konfiguration, die in Teilaufgabe a) an der Kommandozeile übergeben wurde soll nun aus einer Datei gelesen werden. In der Datei soll pro Zeile nur ein Grafikobjekt spezifiziert werden. Eine Beispieldatei ist unter `/proj/i4oovs/pub/aufgabe3/drawing.drw` zu finden. Um eine Zeichnung laden zu können muss die Klasse `ShapeContainerImpl` folgende Methode bereitstellen:

- `public void load(String filename) throws IOException, FormatException;`
  Diese Methode soll eine `IOException` werfen wenn während des Lesens ein Fehler auftritt. Falls während der Erzeugung der Objekte oder der Konvertierung der Positionen in ein `int` ein Fehler auftritt, so soll eine `FormatException` geworfen werden. Die Klasse `FormatException` muss implementiert werden. Die Klasse `IOException` soll von `java.io` importiert werden. Da Klassennamen Unicodezeichen enthalten können soll ein `Reader` verwendet werden um die Datei einzulesen.

e) Analog zu vorhergehenden Teilaufgabe soll die Klasse `ShapeContainerImpl` nun um die Möglichkeit erweitert werden eine Zeichnung abzuspeichern. Folgende Methode wird dazu von der graphischen Benutzeroberfläche aufgerufen:

- `public void save(String filename) throws IOException;`

Bearbeitung: bis zum 10.5.2002