Grundlagen der Programmierung 2 SS 2005 - Aufgabenblatt 2

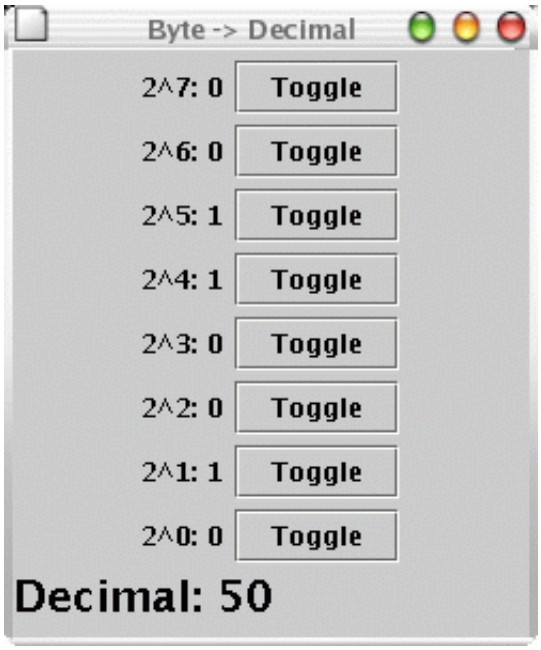
Ausgabe: 22.04.05

Aufgabe 5 (Wiederholung Layout-Manager)

Wie verhalten sich die drei in der Vorlesung vorgestellten Layout-Manager bei Veränderungen an der Form des Fensters bezüglich ihrer Invarianz?

Aufgabe 6 (Konverter: Byte nach Dezimalzahl)

Die Umrechnung von achtstelligen Binärzahlen in Dezimaldarstellung liegt jedem Informatiker zwar schon im Blut, dennoch soll hier ein Umrechnungstool mit einer einfachen Benutzeroberfläche zur Verfügung gestellt werden.



Mit den Buttons können die einzelnen Bits invertiert werden, das Ergebnis der Umrechnung wird sofort in der letzten Zeile angezeigt.

a) Realisieren Sie zunächst die Oberfläche. Um Label und Button pro Zeile waarechrecht anzuordnen soll jeweils ein `FlowLayout` mit Einstellung `FlowLayout.CENTER` verwendet werden. Um die Zeilen dann senkrecht anzuordnen, verwenden Sie ein `GridLayout` mit 9 Zeilen.

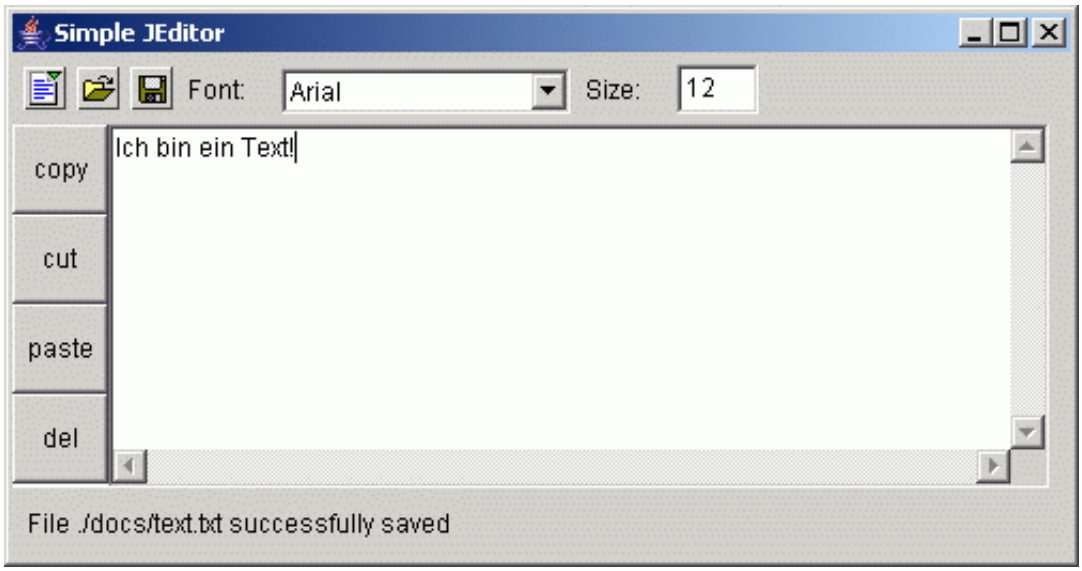
b) Um die Funktionalität zu realisieren, definieren sie am besten pro Bit eine boolesche Variable, die den Zustand (gesetzt oder nicht) anzeigt. Verwenden Sie einen einfachen `ActionListener` wie auf Folie 97a. Sie benötigen nur EINEN! `ActionListener` für alle Buttons.

Aufgabe 7 (Entwurf eines Objektbaums und JavaAPI Recherche)

a) Entwerfen Sie zu der unten dargestellten Benutzungsoberfläche eines einfachen Texteditors den passenden Objektbaum (vergleiche Folie 94). Im Objektbaum sollen die verwendeten Komponenten, Klassen-Namen, Variablen-Namen und Layout-Manager angegeben werden.

Hinweise: Die Buttons mit Bildsymbol wurden sind normale `JButton` Komponenten, denen ein Bildsymbol in folgender Weise zugewiesen wurde:

```
ImageIcon icon = new ImageIcon("image.gif");
JButton myButton = new JButton(icon);
```



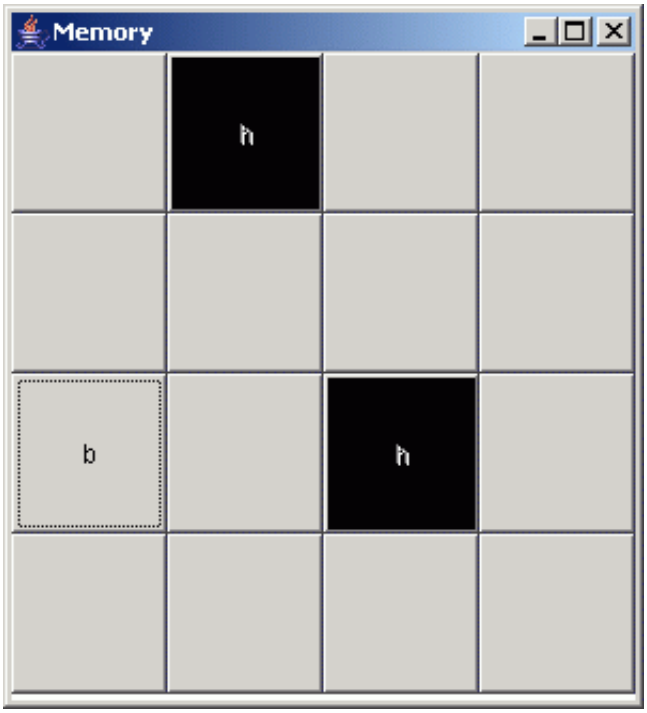
Schreiben Sie ein Programm, das den gezeigten Texteditor aus Komponenten zusammensetzt und darstellt, ohne jedoch den einzelnen Bestandteilen der Benutzeroberfläche Funktionalitäten zuzuordnen. Die Bilder für die Buttons werden im Verzeichnis `images` bereitgestellt.

b) Nutzen Sie die JavaAPI und weiterführende Literatur zum Lösen des folgenden Aufgabenteils:

- Welche beiden Methoden liefern die bevorzugte Größe und die minimale Größe einer Komponente?
- Welche beiden Methoden aus der Schnittstelle `LayoutManager` müssen implementiert werden, damit ein Layout-Manager bei der Berechnung die bevorzugte und minimale Größe eines Behälters berücksichtigt?
- Vorhandene Layout-Manager berücksichtigen die Größe der Komponenten auf verschiedene Art und Weise. Entweder sie ignorieren sie vollständig, sie ignorieren sie nur teilweise oder sie berücksichtigen sie vollständig. Finden Sie heraus, wie sich die drei in der Vorlesung vorgestellten Layout-Manager verhalten?

Aufgabe 8 (Verwendung von Komponenten und Layout-Managern)

In dieser Aufgabe soll ein "Memory"-Spiel implementiert werden. Dabei sollen die Karten als dynamische Buttons unter Verwendung eines sinnvoll gewählten Layout-Managers generiert werden. Das Drücken eines Buttons simuliert das Wenden einer Karte. Gewendete Karten werden mit einem Buchstaben oder einer Zahl beschriftet. Wurden zwei Karten mit gleichem Wert aufgedeckt, so werden diese beiden Buttons gesperrt (`button.setEnabled(false)`). Sind die Werte der gewendeten Karten ungleich, so werden die Karten wieder zugedeckt, bzw. die Beschriftung der Buttons wieder gelöscht. Sind alle Karten gewendet, bzw. alle Buttons gesperrt, so ist das Spiel zu Ende.



Kommentieren Sie Ihren Quellcode mit kurzen und sinnvollen Kommentaren.