Projektblatt P9 (25 P)

Abgabe: Freitag 17. November 2023, 12:00h

Entpacken Sie zunächst die Archiv-Datei `vorgaben-p09.zip`, in der sich die Rahmendateien für die zu lösenden Aufgaben sowie ein Unterverzeichnis mit Bildern befinden. Ergänzen Sie die Dateien durch Ihre Lösungen gemäß der Aufgabenstellung unten. Der hinzuzufügende Java-Sourcecode sollte **syntaktisch richtig** und **vollständig formatiert** sein. Alle Dateien sollten am Ende fehlerfrei übersetzt werden können.
Verpacken Sie die `.java` Dateien für Ihre Abgabe in einem ZIP-Archiv mit dem Namen `IhrNachname.IhrVorname.P09.zip`, welches Sie auf Ilias hochladen.
Führen Sie dazu in dem Verzeichnis, in dem Sie die Dateien bearbeitet haben, folgenden Befehl auf der Kommandozeile aus:

```
zip  IhrNachname.IhrVorname.P09.zip *.java
```

Aufgabe 1: Swing Komponenten und Layout 10 Punkte

In dieser Aufgabe sollen Sie eine Oberfläche entwickeln, die (später) für ein einfaches *Memory* Spiel genutzt werden soll. Dabei geht es darum, Paare von Karten aufzudecken, die das gleiche Bild zeigen. Beginnend mit einer Menge von verdeckten Karten, deckt der Spieler in jedem Zug jeweils zwei Karten auf: sind diese gleich, werden die Karten vom Spielfeld entfernt. Ist dies nicht der Fall, werden die beiden aufgedeckten Karten wieder umgedreht.
Hierzu ist folgendes vorgegeben:

- In dem Unterverzeichnis `bilder`, welches Sie nach dem Auspacken des Archivs vorfinden sollten, liegen 15 Bilder, die für die Bilder auf den Memory-Karten verwendet werden sollen.

- Die Klasse `Karte` repräsentiert eine Memory-Karte, deren Attribute aus einem Bild (für die Vorderseite der Karte) und einem Feld von zwei Karten-Positionen bestehen. Die Objekte, die die beiden Positionen der Karte auf einem rechteckigen Raster festlegen, sind Instanzen der Klasse `Position`, welche als innere Klasse realisiert ist. Die Klasse `Position` besitzt zwei Attribute `zeile` und `spalte`, die durch den Konstruktor initialisiert und durch die zugehörigen Getter-Methoden aus einem Objekt ausgelesen werden können.

- In der Klasse `Resources` wird eine Liste aller (durch die Bilddateien verfügbaren) Memory-Karten definiert und durch statischen Code initialisiert. Die Klassenmethode `initFeld` initialisiert ein als Parameter übergebenes Feld mit einer zufälligen Verteilung der Karten. Hierzu referenzieren jeweils zwei zufällig ausgewählte Positionen auf dem Feld $(zeile_1, spalte_1)$ und $(zeile_2, spalte_2)$ eine der verfügbaren Karten.

- Die Klasse `KartenButton` ist von der Swing-Klasse `JToggleButton` abgeleitet und repräsentiert eine Menory-Karte mit zwei Zuständen (Button selektiert = Karte aufgedeckt, Button nicht selektiert = Karte verdeckt). In der von der Oberklasse geerbten und überschriebenen `paint`-Methode wird, abhängig vom Zustand des Buttons, die Vorder- bzw. Rückseite der Karte gezeichnet. Ist die Karte über das Attribut `karteEntfernt` gekennzeichnet, wird der Bereich für die Karte komplett weiß eingefärbt.

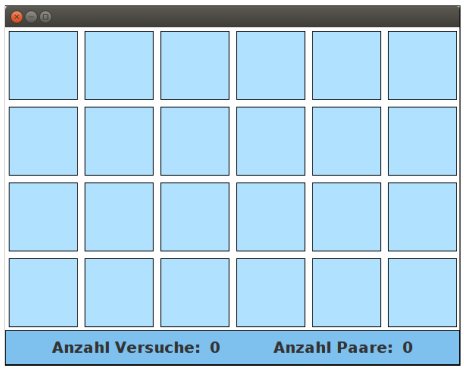
- Die Klasse `ZaehlerLabel` ist von der `Swing`-Klasse `JLabel` abgeleitet und repräsentiert ein Label, dass eine ganze Zahl anzeigt. Initial ist dieser Wert 0 und kann danach mit der Methode `increase` jeweils um einen ganzzahligen Wert `delta` verändert werden. Die momentan noch fast leere Klasse `MyMemory` definiert einen Rahmen für das Spiel.

Sie sollen nun zunächst die grafische Oberfläche erstellen. Die Spielelogik und die Interaktion mit dem User ist Bestandteil des nächsten Übungsblatts.
Ergänzen Sie hierzu nun die Klasse `MyMemory` wie im Folgenden beschrieben:

- Die Oberfläche soll aus zwei Bereichen bestehen: Im oberen (zentralen) Bereich soll ein Gitter von `anzZeilen` × `anzSpalten` angezeigt werden, welches mit einer entsprechenden Anzahl von Instanzen der Klasse `KartenButton` gefüllt ist. Initialiseren Sie den Button in der i. Zeile und der j. Spalte mit der Karte `kartenfeld[i][j]`. Die Variable `kartenfeld` ist dabei als Attribut der Klasse vorgegeben. Der Bereich soll eine weiße Hintergrundfarbe besitzen.

- Im unteren Bereich der Oberfläche soll (später) angezeigt werden, wieviele Versuche (ein Versuch = zwei ausgewählte Karten umdrehen) der Spieler bereits unternommen hat und wieviele Paare er dabei gefunden hat. Verwenden Sie hierzu zwei Labels mit reinem Text und zwei Instanzen der Klasse `ZaehlerLabel`. Jeweils eines der beiden `ZaehlerLabel`s soll mit dem dazugehörigen Label für den Text von dem anderen `ZaehlerLabel` und dem zweiten Textlabel durch einen Abstand getrennt sein. Alle Komponenten im unteren Bereich sollen mit einem farbigen (zu schwarz ausreichend im Kontrast stehenden) Hintergrund ausgestattet sein. Der untere Bereich soll einen einfachen, schwarzen Rahmen bseitzen. Die Labels sollen mit einer größeren Schrift als dem Default-Font ausgestattet werden.

Sie können sich bei der Implementierung an dem folgenden Screenshots orientieren, die ein Beispiel der Oberfläche mit allen verdeckten bzw. allen aufgedeckten Karten zeigt. Die aufgedeckten Karten sollten Sie sehen, wenn Sie in der Klasse KartenButton die Zeile 41 (`if (this.isSelected()){` durch die Zeile `if (!this.isSelected()){` ersetzen.



Aufgabe 2: Swing Komponenten und Layout 15 Punkte

In dieser Aufgabe sollen sie eine Oberfläche entwickeln, welche Komponenten bereitstellt, mit der Termine (bestehend aus einem Datum, einer Uhrzeit und einem Titel) festgelegt werden können. Alle Termine sollen dann bei Bedarf in einer Liste angezeigt werden können. Die Grundelemente, die die GUI enthalten soll, bestehen aus

- einer Eingabemöglichkeit für ein Datum (Tag, Monat und Jahr)
- einer Eingabemöglichkeit für eine Uhrzeit (Stunden und Minuten)
- einer Eingabemöglichkeit für einen kurzen Text, der den Termin inhaltlich beschreibt.
- einem Anzeigebereich für einen längeren Text, in dem später die Terminliste gezeigt wird.

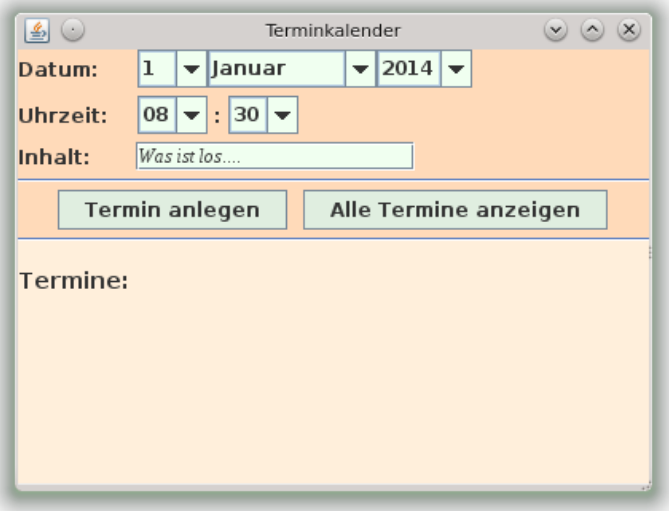
Sie sollen hier zunächst nur die Oberfächengestaltung implementieren, die Umsetzung der Interaktion mit diesen Komponenten wird Bestandteil des nächsten Übungsblatts sein. Folgende Rahmenbedingungen müssen bei der Entwicklung eingehalten werden:

- Für die Eingabe der einzelnen Datumsbestandteile (Jahr, Monat, Tag, Stunde, Minute) sollen Instanzen der Klasse `JComboBox` verwendet werden. Stunden und Minuten sollen dabei jeweils zweistellig dargestellt werden Hinweis: Verwenden sie eine entsprechende Formatierer-Klasse,

um die `String`-Einträge (für Zahlenwerte wie Tag, Stunde oder Minute) in der jeweiligen `JComboBox`-Instanz zu generieren. Die möglichen Werte für Tage sollen aus dem Intervall $[1, 31]$ stammen, die Werte für die Jahre sollen aus die Werte der Menge $\{2023, 2024, 2025\}$ beschränkt werden.

- Die Beschreibung des Inhalts eines Termins soll durch eine `JTextField` Komponente ermöglicht werden und der Bereich, in dem später die Termine angezeigt werden sollen, soll durch eine `JTextArea` Instanz umgesetzt werden.
- Verwenden Sie `JLabel` Instanzen, um auf der grafischen Oberfläche die einzelnen zu spezifizierenden Bestandteile eines Termins zu kennzeichnen.
- Die Oberfläche soll zwei Buttons beinhalten, mit denen später ein Termin generiert und in einer Liste gespeichert werden kann.
- Die Komponenten sollen mit Hilfe verschiedener Container und Layout Manager strukturiert, übersichtlich und zweckgebunden dargestellt werden. Verwenden Sie mindestens drei verschiedene (Hintergrund-)Farben und zwei verschiedene Fonts bei der Gestaltung der Oberfläche.
- Die Klasse `TerminListe` gibt eine Grundstruktur vor, die Ihnen dabei helfen soll, Ihre Implementierung lesbar und übersichtlich zu gestalten.

Bei der Lösung der Aufgabe sollten Sie sich an folgendem Beispiel orientieren. Sie dürfen das Design aber gerne verbessern.


Terminkalender
Datum:
1
Januar
2014
Uhrzeit:
08
:
30
Inhalt:
Was ist los....
Termin anlegen
Alle Termine anzeigen
Termine: