Java08 − Schachbrett komplett

Projekt-Beschreibung: Schachbrett

Wir wollen als BlueJ-Projekt ein Schachbrett zeichnen lassen.

Als Vorlage verwenden wir dazu das BlueJ-Projekt zum Zeichnen von Objekten (aus Jgst. 9).

Als Vorübung erstellen wir eine Klasse „Reihe“ als eindimensionales Array von Quadraten, mit der eine (einzige) Schachbrett-Reihe gezeichnet werden kann.

Dann erstellen wir eine Klasse „Schachbrett“ als zweidimensionales Array von Quadraten, mit der das zweidimensionale Schachbrett gezeichnet werden kann.

Schritt 0 − Vorüberlegungen

Ein Schachbrett hat 8 · 8 = 64 Felder. Man könnte ein Schachbrett als eindimensionales Array der Länge 64 mit 64 Quadraten als Feld-Elementen modellieren.

Dadurch wäre aber die Identifizierung eines bestimmten Feldes schwierig. Wo läge dann das Schachbrett-Feld mit dem Feld-Index 52 (das 53. Feld-Element)?
[Mögliche Lage: (52 + 1) : 8 = 6 Rest 5
→ 6 volle Zeilen, also: 7. Zeile, dort das 5. Feld]

Besser eignet sich zur Modellierung des Schachbretts ein zweidimensionales Array: **`Quadrat[][]`**

Es handelt sich hierbei um ein Array (zweites Paar `[]`) eines Arrays (erstes Paar `[]`) von Quadraten.

Wie erfolgt hier sinnvoll die Identifizierung eines bestimmten Feldes?

Bei **`Quadrat[i][j]`** gibt der Index i die Zeile an und der Index j die Spalte.

i = 0 j = 0	i = 0 j = 1	i = 0 j = 2	…	…	…	…	i = 0 j = 7
i = 1 j = 0	i = 1 j = 1	i = 1 j = 2	…	…	…	…	i = 1 j = 7
i = 2 j = 0	i = 2 j = 1	i = 2 j = 2	…	…	…	…	i = 2 j = 7
…	…	…	…	…	…	…	…
…	…	…	…	…	…	…	…
…	…	…	…	…	…	…	…
…	…	…	…	…	…	…	…
i = 7 j = 0	i = 7 j = 1	i = 7 j = 2	…	…	…	…	i = 7 j = 7

Schritt 1 − Klasse „Schachbrett“ erstellen

Lege im BlueJ-Projekt „Java_04_Schachbrett“ eine neue Klasse „Schachbrett“ an, öffne den Quelltext und lösche den gesamten „automatisch“ erzeugten Quelltext.

Als Quellcode der Klasse „Schachbrett“ können wir den Quellcode der Klasse „Reihe“ übernehmen, wir müssen ihn nur ein wenig ändern und ergänzen (für die „2. Dimension“):

```
public class Schachbrett {
    private Quadrat[][] sb;     // Deklaration „sb“ als 2D-Array v. Quadraten

    public Schachbrett() {     // Konstruktor
        sb = new Quadrat[8][8];     // Initialisierung als 2D-Feld „8 x 8“
        for (int i = 0; i < 8; i++) {      // äußere for-Schleife für Zeilen
            for (int j = 0; j < 8; j++){  // innere for-Schl. für jede Reihe
                sb[i][j] = new Quadrat();   // jedes Feld-Element ein Quadrat
            }
        }     // schließende Klammer der äußeren for-Schleife
    }

    public void bildMalen() {     // Methode zum Malen des Bildes (Leinwand)
        for (int i = 0; i < 8; i++) {      // äußere for-Schleife für Zeilen
            for (int j = 0; j < 8; j++){
                sb[i][j].sichtbarMachen();
            }
        }     // schließende Klammer der äußeren for-Schleife
    }
}
```

Übersetze die Klasse, erzeuge ein neues Objekt „schachbr1“ dieser Klasse „Schachbrett“ und rufe die Methode „`bildMalen()`“ auf.

Man sieht wieder nur ein einzelnes rotes Quadrat, eigentlich sind es 64 „übereinander liegende“ Quadrate.

Der Konstruktor muss also wieder „ausgebaut“ werden (wie bei der Klasse „Reihe“).

Schritt 2 − Ausbau des Konstruktors

Beim Aufruf des Konstruktors (d. h. bei der Erzeugung eines neuen Objekts der Klasse „Schachbrett“) wird ein Feld aus 8 x 8 Quadraten erzeugt. Jedes Feld-Element (d. h. jedes Quadrat) wird mit den Standard-Werten für die Erzeugung eines Quadrats versehen (zu finden im Quellcode der Klasse „Quadrat“).

Was ist zu tun, damit ein komplettes Schachbrett aus 8 x 8 = 64 Quadraten erzeugt wird?

a) Größe der Quadrate ändern (für alle Quadrate, unabhängig von den Feld-Indizes i und j)
b) Position der Quadrate ändern (in Abhängigkeit von den Feld-Indizes i und j)
c) Farbe der Quadrate ändern (in Abhängigkeit von den Feld-Indizes i und j)

zu a) Größe der Quadrate ändern

Hierzu rufen wir für jedes Feld-Element „`sb[i][j]`“ (d. h. für jedes Quadrat des Schachbretts) die Methode „`groesseAendern(int neueGroesse)`“ (der Klasse Quadrat) auf und übergeben als „neueGroesse“ den Wert „30“, damit die Quadrate eine Seitenlänge von 30 Pixeln erhalten.

Im Quellcode des Konstruktors muss die blaue Zeile eingefügt werden (innerhalb der inneren for-Schleife):

```
sb[i][j] = new Quadrat();
sb[i][j].groesseAendern(30);
```

zu b) Position der Quadrate ändern

Hierzu rufen wir für jedes Feld-Element „`sb[i][j]`“ (d. h. für jedes Quadrat des Schachbretts) die Methode „`positionSetzen(int xPos, int yPos)`“ (der Klasse Quadrat) auf.

Die y-Position (d. h. die Zeile) muss nun (im Gegensatz zu „Reihe“) unterschiedlich sein. Sie hängt vom Feld-Index i ab und soll den Wert „`30 + 32*i`“ bekommen (2 Pixel Abstand zwischen den Quadraten).

Die x-Position (d. h. die Spalte) muss weiterhin (wie bei „Reihe“) unterschiedlich sein. Sie hängt vom Feld-Index j ab und soll den Wert „`120 + 32*j`“ erhalten (2 Pixel Abstand zwischen den Quadraten).

Füge den hierfür notwendigen Code (unterhalb von **`sb[i][j].groesseAendern(30);`**) in den Quellcode des Konstruktors ein (d. h. innerhalb der inneren for-Schleife).

Übersetze die Klasse, erzeuge ein neues Objekt „schachbr1“ der Klasse „Schachbrett“ und rufe die Methode „`bildMalen()`“ auf.

Was sieht man nun auf der „Leinwand“? (Wow!)

zu c) Farbe der Quadrate ändern

Für i = 0 und j = 0 soll das Quadrat die Farbe „schwarz“ erhalten, für i = 0 und j = 1 die Farbe „gelb“ usw.

Betrachte das Schachbrett-Modell oben! Was haben die Indizes i und j der „schwarzen“ Felder gemeinsam?

Zur Unterscheidung, welche Quadrate welche Farbe erhalten, nutzen wir wieder die „modulo-Funktion“.
Die „modulo-Funktion“ „a % b“ berechnet den Rest bei der Division von „a“ durch „b“. Also:

„x % 2“ liefert den Wert „0“, wenn x eine gerade Zahl ist, und den Wert „1“, wenn x eine ungerade Zahl ist.

Nötig ist also eine bedingte Anweisung („`if (…)`“) mit einer passenden Bedingung. (Was ist x hier?)

Wenn die Bedingung erfüllt ist, soll mit der Methode „`farbeAendern(String neueFarbe)`“ (der Klasse „Quadrat“) das entsprechende Quadrat die Farbe „schwarz“ erhalten, ansonsten die Farbe „gelb“.

Füge den hierfür notwendigen Code (unterhalb von **`sb[i][j].positionsSetzen(…);`**) in den Quellcode des Konstruktors ein.

Übersetze die Klasse, erzeuge ein neues Objekt „schachbr1“ der Klasse „Schachbrett“ und rufe die Methode „`bildMalen()`“ auf. Was sieht man nun auf der „Leinwand“?

Wenn ein schwarz-gelbes Schachbrett zu sehen ist: Herzlichen Glückwunsch!