

5.6.2022 20:27:21

Automat.java

Page 1/2

```

1 package warenautomat;
2
3 import java.time.LocalDate;
4
5 /**
6  * Der Automat besteht aus 7 Drehtellern welche wiederum je aus 16 Fächer
7  * bestehen. <br>
8  * Der erste Drehteller und das jeweils erste Fach haben jeweils die Nummer 1
9  * (nicht 0!). <br>
10 * Im Weiteren hat der Automat eine Kasse. Diese wird vom Automaten instanziiert.
11 */
12 public class Automat {
13
14     private static final int NR_DREHTELLER = 7;
15     private Drehteller[] drehteller;
16     private Kasse kasse;
17
18     /**
19      * Der Standard-Konstruktor. <br>
20      * Führt die nötigen Initialisierungen durch (u.a. wird darin die Kasse
21      * instanziiert).
22      */
23     public Automat() {
24
25         // TODO
26
27     }
28
29     /**
30      * Füllt ein Fach mit Ware. <br>
31      * Wenn das Service-Personal den Automaten füllt, wird mit einem
32      * Bar-Code-Leser zuerst die Ware gescannt. <br>
33      * Daraufhin wird die Schiebe-Tür geöffnet. <br>
34      * Das Service-Personal legt die neue Ware ins Fach und schliesst das Fach. <br>
35      * Die Hardware resp. System-Software ruft die Methode
36      * <code>Automat.neueWareVonBarcodeLeser() </code> auf.
37      *
38      * @param pDrehtellerNr Der Drehteller bei welchem das Fach hinter der
39      * Schiebe-Türe gefüllt wird. <br>
40      * Nummerierung beginnt mit 1 (nicht 0)!
41      * @param pWareName Der Name der neuen Ware.
42      * @param pPreis Der Preis der neuen Ware.
43      * @param pVerfallsDatum Das Verfallsdatum der neuen Ware.
44      */
45     public void neueWareVonBarcodeLeser(int pDrehtellerNr, String pWareName,
46                                         double pPreis, LocalDate pVerfallsDatum) {
47
48         // TODO
49
50     }
51
52     /**
53      * Gibt die Objekt-Referenz auf die <em> Kasse </em> zurück.
54      */
55     public Kasse gibKasse() {
56         return kasse;
57     }
58
59     /**
60      * Wird von der System-Software jedesmal aufgerufen wenn der gelbe Dreh-Knopf
61      * gedrückt wird. <br>
62      * Die Applikations-Software führt die Drehteller-Anzeigen nach (Warenpreis,
63      * Verfallsdatum). <br>
64      * Das Ansteuern des Drehteller-Motors übernimmt die System-Software (muss
65      * nicht von der Applikations-Software gesteuert werden.). <br>
66      * Die System-Software stellt sicher, dass <em> drehen </em> nicht durchgeführt wird
67      * wenn ein Fach offen ist.
68      */
69     public void drehen() {
70
71         // TODO

```

5.6.2022 20:27:21

Automat.java

Page 2/2

```

72
73     }
74
75     /**
76      * Beim Versuch eine Schiebetüre zu öffnen ruft die System-Software die
77      * Methode <code>oeffnen() </code> der Klasse <em> Automat </em> mit der
78      * Drehteller-Nummer als Parameter auf. <br>
79      * Es wird überprüft ob alles o.k. ist: <br>
80      * - Fach nicht leer <br>
81      * - Verfallsdatum noch nicht erreicht <br>
82      * - genug Geld eingeworfen <br>
83      * - genug Wechselgeld vorhanden <br>
84      * Wenn nicht genug Geld eingeworfen wurde, wird dies mit
85      * <code>SystemSoftware.zeigeZuWenigGeldAn() </code> signalisiert. <br>
86      * Wenn nicht genug Wechselgeld vorhanden ist wird dies mit
87      * <code>SystemSoftware.zeigeZuWenigWechselGeldAn() </code> signalisiert. <br>
88      * Wenn o.k. wird entriegelt (<code>SystemSoftware.entriegeln() </code>) und
89      * positives Resultat zurückgegeben, sonst negatives Resultat. <br>
90      * Es wird von der System-Software sichergestellt, dass zu einem bestimmten
91      * Zeitpunkt nur eine Schiebetüre offen sein kann.
92      *
93      * @param pDrehtellerNr Der Drehteller bei welchem versucht wird die
94      * Schiebe-Türe zu öffnen. <br>
95      * Nummerierung beginnt mit 1 (nicht 0)!
96      * @return Wenn alles o.k. <code> true </code>, sonst <code> false </code>.
97      */
98     public boolean oeffnen(int pDrehtellerNr) {
99
100         return false; // TODO
101
102     }
103
104     /**
105      * Gibt den aktuellen Wert aller im Automaten enthaltenen Waren in Franken
106      * zurück. <br>
107      * Analyse: <br>
108      * Abgeleitetes Attribut. <br>
109      *
110      * @return Der totale Warenwert des Automaten.
111      */
112     public double gibTotalenWarenWert() {
113
114         return 0.0; // TODO
115
116     }
117
118     /**
119      * Gibt die Anzahl der verkauften Ware <em> pName </em> seit (>=)
120      * <em> pDatum </em> zurück.
121      *
122      * @param pName Der Name der Ware nach welcher gesucht werden soll.
123      * @param pDatum Das Datum seit welchem gesucht werden soll.
124      * @return Anzahl verkaufter Waren.
125      */
126     public int gibVerkaufsStatistik(String pName, LocalDate pDatum) {
127
128         return 0; // TODO
129
130     }
131
132 }

```

5.6.2022 20:27:21

Kasse.java

Page 1/2

```

1 package warenautomat;
2
3
4 import warenautomat.SystemSoftware;
5
6 /**
7  * Die Kasse verwaltet das eingenommene Geld sowie das Wechselgeld. <br>
8  * Die Kasse hat fünf Münz-Säulen für: <br>
9  * - 10 Rappen <br>
10 * - 20 Rappen <br>
11 * - 50 Rappen <br>
12 * - 1 Franken <br>
13 * - 2 Franken <br>
14 */
15 public class Kasse {
16
17     /**
18      * Standard-Konstruktor. <br>
19      * Führt die nötigen Initialisierungen durch.
20      */
21     public Kasse() {
22
23         // TODO
24
25     }
26
27     /**
28      * Diese Methode wird aufgerufen nachdem das Personal beim Verwalten des
29      * Wechselgeldbestand die Münzart und die Anzahl der Münzen über die
30      * Tastatur eingegeben hat
31      * (siehe Use-Case "Wechselgeldbestand (Münzbestand) verwalten").
32      *
33      * @param pMuenzenBetrag Der Betrag der Münzart in Franken.
34      * @param pAnzahl Die Anzahl der Münzen. Bei der Entnahme von Münzen als
35      *     entsprechender negativer Wert.
36      * @return Anzahl der Münzen welche hinzugefügt resp. entnommen werden (bei
37      *     Entnahme als negativer Wert). <br>
38      *     Im Normalfall entspricht dieser Wert dem Übergabeparameter
39      *     <code>pAnzahl</code>. <br>
40      *     Er kann kleiner sein falls beim Hinzufügen in der Münzsäule zu
41      *     wenig Platz vorhanden ist oder wenn bei der Entnahme ein grössere
42      *     Anzahl angegeben wurde als tatsächlich in der Münzsäule vorhanden
43      *     ist. <br>
44      *     Wenn ein nicht unterstützter Münzbetrag übergeben wurde: -200
45      */
46     public int verwalteMuenzbestand(double pMuenzenBetrag, int pAnzahl) {
47
48         return 0; // TODO
49
50     }
51
52     /**
53      * Diese Methode wird aufgerufen nachdem das Personal beim Geldauffüllen den
54      * Knopf "Bestätigen" gedrückt hat
55      * (siehe Use-Case "Wechselgeldbestand (Münzbestand) verwalten"). <br>
56      * Verbucht die Münzen gemäss dem vorangegangenen Aufruf der Methode
57      * <code>verwalteMuenzbestand()</code>.
58      */
59     public void verwalteMuenzbestandBestaetigung() {
60
61         // TODO
62
63     }
64

```

5.6.2022 20:27:21

Kasse.java

Page 2/2

```

65 /**
66  * Diese Methode wird aufgerufen wenn ein Kunde eine Münze eingeworfen hat. <br>
67  * Führt den eingenommenen Betrag entsprechend nach. <br>
68  * Stellt den nach dem Einwerfen vorhandenen Betrag im Kassen-Display dar. <br>
69  * Eingenommenes Geld steht sofort als Wechselgeld zur Verfügung. <br>
70  * Die Münzen werden von der Hardware-Kasse auf Falschgeld, Fremdwährung und
71  * nicht unterstützte Münzarten geprüft, d.h. diese Methode wird nur
72  * aufgerufen wenn ein Münzeinwurf soweit erfolgreich war. <br>
73  * Ist die Münzsäule voll (d.h. 100 Münzen waren vor dem Einwurf bereits darin
74  * enthalten), so wird mittels
75  * <code> SystemSoftware.auswerfenWechselGeld() </code> unmittelbar ein
76  * entsprechender Münz-Auswurf ausgeführt. <br>
77  * Hinweis: eine Hardware-Münzsäule hat jeweils effektiv Platz für 101 Münzen.
78  */
79  * @param pMuenzenBetrag Der Betrag der neu eingeworfenen Münze in Franken.
80  * @return <code> true </code>, wenn er Einwurf erfolgreich war. <br>
81  *     <code> false </code>, wenn Münzsäule bereits voll war.
82  */
83  public boolean einnehmen(double pMuenzenBetrag) {
84
85      return false; // TODO
86
87  }
88
89  /**
90  * Bewirkt den Auswurf des Restbetrages.
91  */
92  public void gibWechselGeld() {
93
94      // TODO
95
96  }
97
98  /**
99  * Gibt den Gesamtbetrag der bisher verkauften Waren zurück. <br>
100 * Analyse: Abgeleitetes Attribut.
101 *
102 * @return Gesamtbetrag der bisher verkauften Waren.
103 */
104 public double gibBetragVerkaufteWaren() {
105
106     return 0.0; // TODO
107
108 }
109
110 }

```

5.6.2022 20:27:21

Drehteller.java

Page 1/1

```
1
2 package warenautomat;
3
4 import java.time.LocalDate;
5
6 import warenautomat.SystemSoftware;
7
8
9 public class Drehteller {
10
11     // TODO
12
13 }
```