Beispiel für saubere Klassentrennung: Verwaltung eines Fuhrparks mit max. 25 Autos

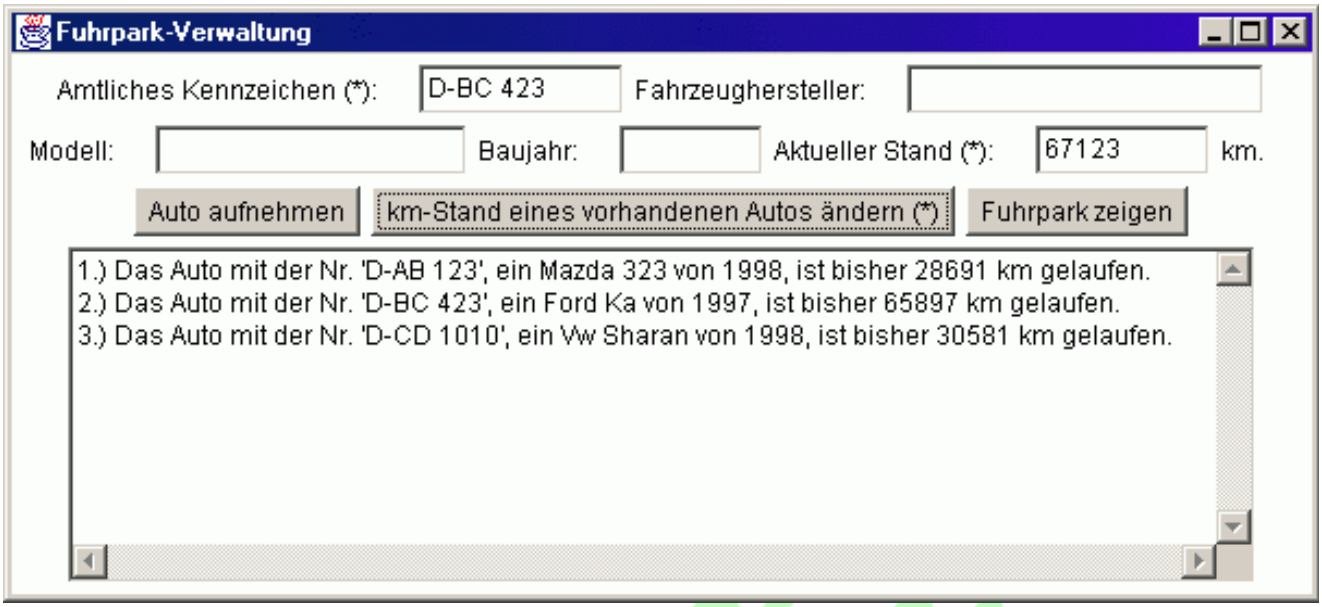
```
1  //Kfz-Verwaltung, Datei 1/4
2  //Java JDK 1.1.18 -- Krell 21.2.2002
3  public class Kfz
4  {
5    //Hier stehen die Daten, die ein (jedes) Auto beschreiben.
6    //Das vorangestellte private sorgt dafür, dass nur die Methoden
7    //dieser Klasse auf die Daten zugreifenb dürfen (verhindert Fehler)
8    private String kennzeichen;      // z.B. "D-AB 123"
9    private String hersteller;       // z.B. "VW"
10   private String typ;              // z.B. "Polo"
11   private int baujahr;             // z.B. 1998
12   private int kmStand;             // z.B. 42389
13
14   //Und hier kommen die Methoden, um die Daten zu ändern oder anzusehen:
15
16   public void merken (String nummer, String firma, String modell,
17     int jahr, int fahrtstrecke)
18     //Beim Aufruf auto.merken ("D-CD 234","Opel","Astra",2001,10893)
19     //werden die übergebenen Daten in die fünf Variablen eingetragen:
20     {
21       kennzeichen = nummer;
22       hersteller     = firma;
23       typ            = modell;
24       baujahr        = jahr;
25       kmStand        = fahrtstrecke;
26     }
27
28   public String nennen ()
29     //Gibt eine Zeichenkette mit allen Daten des Autos aus
30     {
31       String daten = "Das Auto mit der Nr. '"+kennzeichen+"', ein "+
32        hersteller+" "+typ+" von "+baujahr+", ist bisher "+kmStand+" km gelaufen.";
33       return (daten);
34     }
35
36   public String nummer()
37     //Nennt nur das amtl. Kennzeichen des Autos, sonst nichts
38     {
39       return (kennzeichen);
40     }
41
42   public int kmZahl ()
43     //Nennt nur den kmStand des Autos, sonst nichts
44     {
45       return (kmStand);
46     }
47
48   public void neuerStand (int km)
49     //Ändert nur den kmStand des Autos, sonst nichts
50     {
51       kmStand = km;
52     }
53 }
54
55
56
```

```
57 //Kfz-Verwaltung, Datei 2/4
58 //Java JDK 1.1.18 -- Krell 21.2.2002
59 import java.io.*;
60
61 //Erzeugt und verwaltet Reihung fuhrpark aus 25 Objekten der Klasse Kfz
62 public class KfzFuhrpark
63 {
64   //Daten des Fuhrparks
65   private Kfz[] fuhrpark = new Kfz[25]; //(bis zu) 25 Autos im Fuhrpark
66   private int anzahlDerAutos = 0; //zunächst ist noch kein Auto bekannt.
67     //Erlaubt ist/sind 0 <= anzahlDerAutos <= 25
68
69   //Methoden zur Verwaltung des Fuhrparks
70
71   public KfzFuhrpark() //Automatische Initialisierung
72   {
73     for (int i=0; i<25; i++)
74     {
75       fuhrpark[i] = new Kfz(); //Erzeugen der 25 Objekte vom Typ Kfz
76     }
77   }
78
79   public int wagenAnzahl () //Sagt, wie viele Autos es gibt
80   {
81     return (anzahlDerAutos);
82   }
83
84   public void neuesAuto (String nummer, String produzent, String modell,
85     int jahr, int kmStand) //Erweitert fuhrpark um das angegebene Auto
86   {
87     fuhrpark[anzahlDerAutos].merken (nummer,produzent,modell,jahr,kmStand);
88       //Speichert Daten im fuhrpark an der indizierten Stelle durch Aufruf
89       //der Kfz-Methode merken. Jede der 25 fuhrpark-Komponenten ist ein Kfz!
90     anzahlDerAutos = anzahlDerAutos + 1;
91       //Durch die Neuaufnahme hat sich die Zahl der Autos um 1 erhöht!
92   }
93
94   public String listeAlle()
95   //Erzeugt Text mit den Daten aller Fahrzeuge: 1 Zeile pro Auto
96   {
97     String ausgabe = "";
98     for (int i=0; i<anzahlDerAutos; i++)
99     {
100      ausgabe = ausgabe+(i+1)+".) "+fuhrpark[i].nennen()+"\n";
101    }                          //nutzt Kfz-Methode
102    return (ausgabe);
103  }
104
105  public void ändereKmStand (String amtlKennzeichen, int neueKm)
106  //Ändert den kmStand des Autos mit dem angeg. Kennzeichen
107  {
108    //Finden des Index i vom Auto mit der gesuchten Nummer
109    int i = anzahlDerAutos - 1; //von hinten beginnen
110    while ((i>=0)&&(! fuhrpark[i].nummer().equals(amtlKennzeichen)))
111    {                          //equals statt == , da versch. Objekte mit gleichem Inhalt
112      i = i-1;
113    }
114    //KmStand ändern
115    if (i>=0) //Auto gefunden -- sonst ist i = -1
116    {
117      fuhrpark[i].neuerStand (neueKm); //Aufruf der Kfz-Methode
118    }
```

```
119   }
120 }
121 //Kfz-Verwaltung, Datei 3/4
122 //Java JDK 1.1.18 -- Krell 21.2.2002
123 import java.awt.*;
124 import java.awt.event.*;
125 import java.io.*;
126
127 //Benutzer-Oberfläche der Kfz-Verwaltung, die ihrerseits ein Objekt
128 //namens alleAutos der Klasse KfzFuhrpark anlegt und verwaltet
129 public class KfzGUI extends Frame
130 {
131   KfzFuhrpark   alleAutos       = new KfzFuhrpark();  //Def. und Erzeugen
132
133   TextField     tfKennzeichen   = new TextField("",10);
134   TextField     tfHersteller    = new TextField("",20);
135   TextField     tfModell        = new TextField("",17);
136   TextField     tfBaujahr       = new TextField("",6);
137   TextField     tfKmStand       = new TextField("",8);
138   Button        btNeuesAuto     = new Button("Auto aufnehmen");
139   Button        btNeueKm        = new Button("km-Stand eines vorhandenen Autos ändern (*)");
140   Button        btZeigen        = new Button("Fuhrpark zeigen");
141   TextArea      taAusgabe       = new TextArea(9,75);
142
143   public void richteFensterEin() // Fenster initalisieren und beschreiben
144   {
145     //WindowsListener hinzufügen, damit Schließknopf funktioniert
146     addWindowListener (
147       new WindowAdapter ()
148       {
149         public void windowClosing (WindowEvent ereignis)
150         {               //ersetzt bisher leere Methode
151           setVisible (false);
152           dispose();
153           System.exit(0);
154         }
155       }
156     ); // runde Klammer vom Windowlistener geschlossen;
157   }
158
159   public void richteKnöpfeEin()
160   {
161             btNeuesAuto.addActionListener (
162               new ActionListener ()
163               {
164                 public void actionPerformed (ActionEvent e)
165                 {
166                   alleAutos.neuesAuto (tfKennzeichen.getText(), tfHersteller.getText(),
167                     tfModell.getText(), Integer.parseInt(tfBaujahr.getText()),
168                     Integer.parseInt(tfKmStand.getText())); //Auto aufnehmen
169                   tfKennzeichen.setText(""); //und Eingabefelder löschen
170                   tfHersteller.setText("");
171                   tfModell.setText("");
172                   tfBaujahr.setText("");
173                   tfKmStand.setText("");
174                 }
175               }); // runde Klammer (von addActionListener)
176
177             btNeueKm.addActionListener (
178               new ActionListener ()
179               {
180                 public void actionPerformed (ActionEvent e)
```

```
181                 {
182                   alleAutos.ändereKmStand (tfKennzeichen.getText(),
183                     Integer.parseInt(tfKmStand.getText())); //km ändern
184                   tfKennzeichen.setText(""); //und Eingabefelder löschen
185                   tfHersteller.setText("");
186                   tfModell.setText("");
187                   tfBaujahr.setText("["+Integer.parseInt(tfKmStand.getText())+"]");
188                   tfKmStand.setText("");
189                 }
190               }); // runde Klammer (von addActionListener)
191
192             btZeigen.addActionListener (
193               new ActionListener ()
194               {
195                 public void actionPerformed (ActionEvent e)
196                 {
197                   taAusgabe.setText( alleAutos.listeAlle() );
198                 }
199               }); // runde Klammer (von addActionListener)
200   }
201
202   public void führeAus ()
203   {
204     setTitle("Fuhrpark-Verwaltung"); // Fenster-Titel
205     setSize (600,280); // Fenstergröße (Breite und Höhe in Pixeln)
206     setLayout (new FlowLayout());
207     richteFensterEin();
208     richteKnöpfeEin();
209     add (new Label("Amtliches Kennzeichen (*):"));
210     add (tfKennzeichen);
211     add (new Label("Fahrzeughersteller:"));
212     add (tfHersteller);
213     add (new Label("Modell:"));
214     add (tfModell);
215     add (new Label("Baujahr:"));
216     add (tfBaujahr);
217     add (new Label("Aktueller Stand (*):"));
218     add (tfKmStand);
219     add (new Label("km."));
220     add (btNeuesAuto);
221     add (btNeueKm);
222     add (btZeigen);
223     add (taAusgabe);
224     setVisible(true);
225   }
226 }
227
228
```

```
229 //Kfz-Verwaltung, Datei 4/4
230 //Java JDK 1.1.18 -- Krell 21.2.2002
231 //Definiert, erzeugt und startet ein Objekt der Klasse KfzGUI
232 public class KfzStart  {
233   public static void main (String[] s) {
234     KfzGUI autoVerw = new KfzGUI();
235     autoVerw.führeAus();
236   }
237 }
```



Aufgaben:

Ergänze den Java-Quelltext für folgende Anforderungen:

a) Kennzeichen können wechseln (nach Umzug): Erweitere das Programm dafür! Wo/wie?
b) Ein anderes Programm (KfzFamilie statt KfzFuhrpark benutzt auch Kfz). Dann werden Autos verkauft: Mami verkauft an Papi; sie selbst bekommt ein neues Auto. Wie können die Daten von Mamis (altem) Auto an Papis Auto weitergegeben werden? Ergänze Kfz um eine geeignete Methode!
c) Ein Auto (erkennbar am Kennzeichen) soll aus dem Fuhrpark entfernt werden können. In der Reihung sollen keine Lücken bleiben!