Typische Java-AWT-Objekte und -Methoden:

```
1  // Demo: Ein-/Ausgabe mit der Java-AWT
2  // Java jdk 1.1.18 -- R. Krell, 8.11.01/28.11.01
3  import java.awt.*;
4  import java.awt.event.*;
5
6  public class JavaEA extends Frame  // Frame ist Fensterklasse
7  {
8    Label ausgabezeile;  // Deklaration der Bildschirmobjekte
9    Button knopf;
10   TextField eingabezeile;
11   TextArea mehrzeiligesTextfenster;
12   int zähler = 0;  // für Beispiel: Knopfdrücke werden gezählt
13
14   private void richteFensterEin() // Fenster initialisieren und beschreiben
15   {
16     //WindowsListener hinzufügen, damit Schließknopf funktioniert
17     addWindowListener (
18       new WindowAdapter ()
19       {
20         public void windowClosing (WindowEvent ereignis)
21         {                  //ersetzt bisher leere Methode
22           setVisible (false);
23           dispose();
24           System.exit(0);
25         }
26       }
27     ); // runde Klammer vom Windowlistener geschlossen;
28
29     setTitle("Das hier ist die Überschrift"); // Fenster mit Titel versehen
30     setSize (420,200); //Fenstergröße (Breite und Höhe in Pixeln) festlegen
31     setLayout (new FlowLayout()); // innere Unterteilung des Fensters,
32     // kann auch später in führeAus() vor den add(..)-Befehlen stehen
33     // ohne besondere Definition ist FlowLayout voreingestellt.
34   }
35
36   private void richteAusgabeEin()
37   {
38     ausgabezeile = new Label("-----------------------------------");
39     // Erzeugen des Ausgabelabels; Größe richtet sich nach Text!
40   }
41
42   private void richteKnopfEin()
43   {
44     // Knöpfe definieren und beschriften (Größe entsprechend Beschriftung)
45     knopf  = new Button ("Knopf-Beschriftung");
46     // oder: .. = new Button(); knopf.setLabel ("Knopf-Beschriftung")
47
48     //Funktion der Knöpfe festlegen
49     knopf.addActionListener (
50       new ActionListener ()
51       {
52         public void actionPerformed (ActionEvent e)
53         {
54           // hier steht der Programmtext, der ausgeführt werden
55           // soll, wenn der Knopf gedrückt wird. z.B.:
56           zähler++;
57           ausgabezeile.setText ("Knopf wurde "+zähler+"x gedrückt");
58         }
59       }); // runde Klammer (von addActionListener)
60    }
61
```

```
62      private void richteEingabezeileEin ()
63      {
64        eingabezeile = new TextField("Hier tippen und <Enter>",30);
65        //30 = Breite für 30 Zeichen
66        // spätere Textänderung mit eingabezeile.setText("Neuer Text");
67
68        eingabezeile.addActionListener (
69          new ActionListener ()
70          {
71            public void actionPerformed (ActionEvent e)
72            {
73              // hier steht der Programmtext, der nach der Eingabe (wenn die
74              // <Eingabe>-Taste gedrückt wurde) ausgeführt werden soll, z.B.:
75              ausgabezeile.setText ("Eingabe '"+eingabezeile.getText()+"'");
76              // Eingaben können wie folgt in Kommazahlen verwandelt werden:[1])
77              Double hilfszahlobjekt = Double.valueOf (eingabezeile.getText());
78              double kommazahl = hilfszahlobjekt.doubleValue();
79              mehrzeiligesTextfenster.append ("Als Dezimalbruch="
80                  +kommazahl+"\n");
81              // oder -- weniger aufwändig -- in Ganzzahlen:
82              int ganzzahl = Integer.parseInt (eingabezeile.getText());
83              mehrzeiligesTextfenster.append ("Ganze Zahl="+ganzzahl+"\n");
84              // Falls die Eingabe gelöscht werden soll:
85              eingabezeile.setText("");
86            }
87          }); // runde Klammer (von addActionListener)
88       }
89
90      private void richteTextfensterEin ()
91      {
92        mehrzeiligesTextfenster = new TextArea("Zeile1\nZeile2\n",4,30);
93        // 4 Zeilen hoch, 30 Zeichen breit. (Anfangs-)Text kann auch später
94        // eingefügt werden mit:
95        // mehrzeiligesTextfenster.setText("Zeile1\nZeile2\nZeile3\n");
96        mehrzeiligesTextfenster.setEditable(false);
97        // bei true kann der Textfensterinhalt auf dem Bildschirm vom Benutzer
98        // verändert werden (Normalfall), bei false nicht.
99        // Üblicherweise kein ActionListener, sondern Button für Aktionen!
100     }
101
102     public void führeAus ()
103     {
104       richteFensterEin();       // Aufruf der Methoden zum Erzeugen der
105       richteKnopfEin();         // Bildschirmobjekte und der Definition
106       richteAusgabeEin();       // der bei Ereignissen auszuführenden Methoden
107       richteEingabezeileEin();
108       richteTextfensterEin();
109
110       add (knopf);              // Anordnung der Objekte auf dem Bildschirm
111       add (ausgabezeile);       // in der hier gewählten Reihenfolge bzw.
112       add (eingabezeile);       // mit dem hier gültigen Layout-Manager
113       add (mehrzeiligesTextfenster);
114       setVisible(true);         // Macht Bildschirm mit allen angeordneten
115         // Objekten sichtbar und startet damit das Programm
116     }
117   }
```

---

[1]) in der neuen Java-JDK bzw. SDK 2.3 gelingt auch bei Kommazahlen vom Typ Float oder Double die Umwandlung wie bei Ganzzahlen ohne Zwischenobjekt (Z. 78/79) z.B. mit double kommazahl = Double.parseDouble (eingabezeile.getText());

Fehler bei der Zahl-Umwandlung (weil z.B. Buchstaben eingetippt wurden) führen hier noch nicht zu einer Reaktion. Der Programmtext nach dem fehlerhaften Umwandlungsversuch innerhalb der actionPerformed-Methode wird ignoriert – sonst passiert nichts.