Datentypen

Der Typ jeder verwendeten Variablen muss angegeben werden.

Folgende Datentypen haben wir kennen gelernt:

```
int         negative und positive ganze Zahlen
double      negative und positive Dezimalzahlen
boolean     Wahrheitswert (true oder false)
char        Zeichen (ASCII-Zeichen)
byte        Zahl zwischen 0 und 255
```

Man kann bei der Deklaration einen Wert der Variablen angeben oder später den Wert zuweisen.
Beide Befehlsfolgen leisten dasselbe:

```
int i;          int i=15;
i=15;
```

Man kann Daten in andere Typen umwandeln, indem man vor einen in Klammern gesetzten Term den neuen Type in runden Klammern setzt.

Beispiel: Die Zahl 14.26 soll in die ganze Zahl 14 umgewandelt und ausgegeben werden

```
System.out.println(""+ (int)(14.26))
```

Achtung: Bei der Umwandlung können sich die Werte ändern (siehe Beispiel).

Timer

Ein Timer wird benutzt, wenn eine Aktion schrittweise nach jeweils einem bestimmten Zeitraum ausgeführt werden soll.

```
import javax.swing.Timer;
---------------------------------------
// Methode, in der der Timer eingerichtet wird
t = new Timer (100, (ActionEvent e) ->
{
      // hier steht die Anwendung
});
t.setRepeats(true);
t.start();
---------------------------------------
// Methode, die z. B. nach einem Tastendruck oder einer
// Mausaktion ausgeführt wird
t.stop();
---------------------------------------
// globale Variablen
Timer t;
```

Mit der import-Zeile wird die Bibliothek für den Timer eingebunden.

Um in verschiedenen Methoden auf den Timer zugreifen zu können, wird das Timer-Objekt global definiert.

In der runden Klammer nach Timer steht die Zahl 100 für die Verzögerung bei der Ausführung. In diesem Fall wird alle 100 ms die Anwendung ausgeführt.

Der Timer kann gestoppt werden, indem z. B. in einer Methode, die nach einem Event ausgeführt wird, der Timer mit dem stop-Befehl abgeschaltet wird.

Schleifen

Wiederholungen bestimmter Programmteile nennt man Schleifen.

Die *for*-Schleife benutzt man, wenn man genau weiß, wie oft ein Teil wiederholt werden soll:

```
for (int i=0; i<=10; i=i+1)
{
      // hier steht der Programmteil, der wiederholt werden
      // soll
}
```

Eine Laufvariable muss definiert werden, hier i als ganze Zahl.
i wird bei jeder Wiederholung um 1 vergrößert (siehe ganz rechts) und wird dabei die Zahlenwerte von 0 bis 10 annehmen.
Der Datentyp der Laufvariable darf z. B. auch vom Typ double sein:

```
for (double a=0.52; a<=8.523; a=a+1.992)
{
      // hier steht der Programmteil, der wiederholt werden
      // soll
}
```

Die *while*-Schleife wird wiederholt, solange eine bestimmte Bedingung erfüllt ist:

```
while (a>23)
{
      // hier steht der Programmteil, der wiederholt werden
      // soll
}
```

Wird auf Gleichheit getestet, so müssen 2 Gleichheitszeichen gesetzt werden:

```
while (a == 23)
{
      // hier steht der Programmteil, der wiederholt werden
      // soll
}
```

Komponenten

Die ausgewählten Komponenten sollte man der besseren Unterscheidbarkeit wegen mit erklärenden Namen versehen und z. B. bei einem Button auch mit einer erklärenden Aufschrift. Man erreicht das durch Rechtsklick auf die Komponente und dann durch "Change Variable Name" bzw. "Edit Text".

OK-Button
Durch Doppelklick auf den Button wird eine Methode erzeugt, in die man den Code eingibt, der bei Klick auf den Button während des Programmablaufs ausgeführt werden soll. Beispiel für Ende-Button:

```
System.exit(0);
```

Text Field
In ein Textfeld kann man während des Programmablaufs Text schreiben bzw. den Text auslesen:

```
jTextField1.setText("Dieser Text erscheint im Textfeld");
System.out.println(jTextField1.getText());
jTextField1.setText("1427");
int i = Integer.parseInt(jTextField1.getText());
System.out.println(""+i);
```

Die letzten 2 Zeilen zeigen, wie man Text in einem Textfeld in eine Integer-Zahl umwandelt.

Panel
Ein Panel lässt sich gut als Fläche zum Zeichnen benutzen. Dazu muss man ein Graphics-Objekt erzeugen, das die Grafikbefehle enthält.

```
Graphics g = jPanel1.getGraphics();
g.setColor(Color.red);
g.drawLine(10,15,213,256);
g.fillrect(67,112,56,25);
```

Slider
Mit einem Schieberegler kann man leicht Werte einstellen. Man legt den minimalen und den maximalen Wert fest und schreibt den aktuellen Wert z. B. in ein Textfeld.
Mit Rechtsklick auf den Slider und Auswahl des Events "stateChanged" wird mit der Maus der aktuelle Wert des Schiebereglers eingestellt.
Wird im Textfeld eine Zahl eingegeben, so ändert das Event "actionPerformed" des Textfeldes den Wert des Schiebereglers.

```
// In der Methode zum Festlegen der Slider-Bedingungen
jSlider1.setMinimum(50);
jSlider1.setMaximum(300);
// In der Methode zum Event "stateChanged" des Sliders
jTextField1.setText(""+jSlider1.getValue());
// In der Methode zum Event "actionPeformed" des Textfeldes
jSlider1.setValue(Integer.parseInt(jTextField1.getText()));
```

Bilder speichern und laden

```
// Importieren der notwendigen Bibliotheken
import java.io.*;
import javax.imageio.*;

// Laden eines Bildes
try
  {
    bild = ImageIO.read(new File("datei.png"));
  }
catch (IOException e)
  {
    // hier kann eine Fehlerbehandlung stehen
  }

// globale Variablen
BufferedImage bild;
```

```
// Importieren der notwendigen Bibliotheken
import java.io.*;
import javax.imageio.*;

// Speichern eines Bildes
try
  {
    ImageIO.write(bild, "png, new File("datei.png"));
  }
catch (IOException e)
  {
    // hier kann eine Fehlerbehandlung stehen
  }

// globale Variablen
BufferedImage bild;
```

Auslesen der Farben eines Bildes

Definieren eines internen Bildes, das die Größe eines Panels besitzt:

```
BufferedImage bild = new BufferedImage(jPanel1.getWidth(),
     jPanel1.getHeight(), BufferedImage.TYPE_INT_RGB);
Graphics gB = bild.getGraphics();
```

Auslesen des Farbwertes, wenn man mit gedrückter linker Maustaste die Maus bewegt (MouseDragged). Die Farbwerte werden in den 3 Textfeldern TFrot, TFgruen und TFblau ausgegeben.

```
private void jPanel1MouseDragged(java.awt.event.MouseEvent evt)
{
      int farbe = bild.getRGB(evt.getX(), evt.getY());
      Color c = new Color(farbe);
      TFrot.setText(""+c.getRed());
      TFgruen.setText(""+c.getGreen()):
      TFblau.setText(""+c.getBlue());
}
```