```java
/**
 * Der Farn von Michael Barnsley
 * Program Farn.java
 * 16.05.1999
 * Aktualsierte Version vom 18.10.2007
 * Float-Typen durch double-Typen ersetzt
 * Punkt-Objekte anstelle von Koordinaten eingesetzt.
 */

import java.awt.*;
import java.applet.Applet;
import java.awt.event.*;

class DPoint {
    // Um Geschwindigkeit zu gewinnen, kann auf die Punktobjekte
    // direkt zugegriffen werden.
    public double x;
    public double y;

    public DPoint(double iX, double iY) {
        x=iX;
        y=iY;
    }
}

 public class Farn1 extends Applet implements ActionListener {
    private Button fein;
    private Button grob;
    private int max = 200;

    DPoint pAlt = new DPoint(0.0, 0.0); // Beginne mit diesen Punktkoordinaten
    private int l;                      // Variable für Zufallszahlen
    private int j = 0;                  // Auswahl der Transformationsgleichungen

    private double ax=13.0;             // Streckungsfaktor horizontal
    private double ay=13.0;             // Streckungsfaktor vertikal
    private double vx=120.0;            // Nulpunktverschiebung horizontal
    private double vy=300.0;            // Nullpunktverschiebung vertikal

    // Die Koeffizienten der Transformations-Matrizen
    private double[] a11 = {0.0,  0.84962, -0.1554,  0.1554};
    private double[] a12 = {0.0,  0.025,    0.235,  -0.235};
    private double[] a21 = {0.0, -0.0255,   0.19583, 0.19583};
    private double[] a22 = {0.17, 0.84962,  0.18648, 0.18648};

    private double[] b1  = {0.0,  0.0,      0.0,     0.0};
    private double[] b2  = {0.0,  3.0,      1.2,     3.0};


    public void init() {
        // Die Button-Objekte werden instanziert und initialisiert
        fein  = new Button("fein");
        grob = new Button("grob");

        fein.setBounds (10,30,100,40);
        add(fein);
        grob.setBounds (10,70,100,40);
        add(grob);
        fein.addActionListener(this);
        grob.addActionListener(this);
    }

    public void actionPerformed(ActionEvent event) {
        // Die Button-Events ändern die Anzahl der Iterationen
          if(event.getSource() == fein) {
            if(max<200000)
                max *=10;
          }
          else {
            if(max>200)
                max/=10;
```

```
72            }
73            repaint();
74      }
75
76
77      // Die Methode Iter multipliziert die Matrix {a11, a12, a21, a22}
78      // mit dem Ausgangsvektor {x, y} und addieren die Matrix {b1, b2}.
79      public void iter(DPoint pAlt, int n) {
80          double xTemp = a11[n] * pAlt.x + a12[n] * pAlt.y + b1[n];
81          double yTemp = a21[n] * pAlt.x + a22[n] * pAlt.y + b2[n];
82          pAlt.x = xTemp;
83          pAlt.y = yTemp;
84      }
85
86      public void paint(Graphics g) {
87           setBackground(Color.white);
88           // Ein dunkles Grün als Zeichenfarbe auswählen
89           g.setColor(new Color(0, 180, 85));
90           // Die Iterationsschleife
91           for (int i = 0; i < max; i++) {
92               // Mit unterschiedlicher Häufigkeit die Transformationen ziehen.
93               l =  (int) (Math.random() * 100);
94               // Häufigkeit = 56 %
95               if(l<=55)
96                   j=1;
97               // Häufigkeit = 27 %
98               else if((l>55) && (l<=83))
99                   j=3;
100              // Häufigkeit = 13 %
101              else if((l>83) && (l<=96))
102                  j=2;
103              // Häufigkeit = 3 %
104              else
105                  j=0;
106
107              iter(pAlt, j);
108              if (i > 100)
109                 g.drawRect((int) (ax * pAlt.x  +vx), (int) (vy - ay * pAlt.y), 0, 0);
110          }
111          g.setColor(Color.BLACK);
112          g.drawString(String.valueOf(max) + " Iterationen abgeschlossen", 50, 320);
113       }
114 }
115
116
```