

```

1 import javafx.application.*;
2 import javafx.scene.*;
3 import javafx.scene.layout.*;
4 import javafx.scene.control.*;
5 import javafx.scene.paint.*;
6 import javafx.scene.image.*;
7 import javafx.scene.effect.*;
8 import javafx.stage.*;
9 import javafx.geometry.*;
10 import javafx.collections.*;
11
12 ///////////////////////////////////////////////////
13 // 問1 下のように同じ画像 (Food.jpg) を4つ表示してください...
14 ///////////////////////////////////////////////////
15
16 public class Assignment5_1 extends Application
17 {
18     public void start(Stage stage) throws Exception
19     {
20         // 4枚の画像を生成/設定します
21         ImageView[] ivs = new ImageView[4];
22         ivs[0] = new ImageView("Food.jpg");
23         ivs[1] = new ImageView("Food.jpg");
24         ivs[2] = new ImageView("Food.jpg");
25         ivs[3] = new ImageView("Food.jpg");
26
27         // エフェクトを生成し、画像に適用します
28         GaussianBlur gb;
29         gb = new GaussianBlur();
30         gb.setRadius(5.0);
31         ivs[1].setEffect(gb);
32         gb = new GaussianBlur();
33         gb.setRadius(10.0);
34         ivs[2].setEffect(gb);
35         gb = new GaussianBlur();
36         gb.setRadius(20.0);
37         ivs[3].setEffect(gb);
38
39         // レイアウトHBoxを生成/設定します
40         HBox hb = new HBox();
41         ObservableList<Node> lst = hb.getChildren();
42         lst.addAll(ivs);
43
44         // シーンを生成/設定します
45         Scene scene = new Scene(hb);
46
47         // ステージを設定します
48         stage.setScene(scene);
49         stage.setTitle("★ぼかしの効果★");
50
51         // ステージを表示します
52         stage.show();
53     }
54
55     public static void main(String[] args)
56     {
57         launch(args);
58     }
59 }
60

```

```

1 import javafx.application.*;
2 import javafx.scene.*;
3 import javafx.scene.layout.*;
4 import javafx.scene.control.*;
5 import javafx.scene.paint.*;
6 import javafx.scene.image.*;
7 import javafx.scene.effect.*;
8 import javafx.stage.*;
9 import javafx.geometry.*;
10 import javafx.collections.*;
11
12 //////////////////////////////////////////////////
13 // 問2 デジタル数字の画像ファイル (Digital.jpg) があります...
14 //////////////////////////////////////////////////
15
16 public class Assignment5_2 extends Application
17 {
18     public void start(Stage stage) throws Exception
19     {
20         // 4枚の画像を生成／設定します
21         ImageView[] ivs = new ImageView[4];
22         ivs[0] = new ImageView("Digital.jpg");
23         ivs[1] = new ImageView("Digital.jpg");
24         ivs[2] = new ImageView("Digital.jpg");
25         ivs[3] = new ImageView("Digital.jpg");
26
27         // ビューポートを生成し、画像に適用します
28         Rectangle2D[] rts = new Rectangle2D[4];
29         rts[0] = new Rectangle2D(114*2, 183*0, 114, 183);
30         rts[1] = new Rectangle2D(114*0, 183*0, 114, 183);
31         rts[2] = new Rectangle2D(114*1, 183*0, 114, 183);
32         rts[3] = new Rectangle2D(114*1, 183*1, 114, 183);
33         for(int i=0;i<ivs.length;i++)
34             ivs[i].setViewport(rts[i]);
35
36         // レイアウトHBoxを生成／設定します
37         HBox hb = new HBox();
38         ObservableList<Node> lst = hb.getChildren();
39         lst.addAll(ivs);
40
41         // シーンを生成／設定します
42         Scene scene = new Scene(hb);
43
44         // ステージを設定します
45         stage.setScene(scene);
46         stage.setTitle("★デジタル数字★");
47
48         // ステージを表示します
49         stage.show();
50     }
51
52     public static void main(String[] args)
53     {
54         launch(args);
55     }
56 }
57

```

```

1 import javafx.application.*;
2 import javafx.scene.*;
3 import javafx.scene.layout.*;
4 import javafx.scene.control.*;
5 import javafx.scene.paint.*;
6 import javafx.scene.image.*;
7 import javafx.scene.effect.*;
8 import javafx.stage.*;
9 import javafx.geometry.*;
10 import javafx.collections.*;
11
12 ///////////////////////////////////////////////////
13 // 問3 下のように4枚の画像 (Food.jpg Cafe.jpg Beach.jpg ...)
14 ///////////////////////////////////////////////////
15
16 public class Assignment5_3 extends Application
17 {
18     public void start(Stage stage) throws Exception
19     {
20         // 4枚の画像を準備します
21         String[] files = {"Food.jpg", "Cafe.jpg", "Beach.jpg", "Sunset.jpg"};
22         Image[] imgs = new Image[files.length];
23         for(int i=0; i<imgs.length; i++)
24             imgs[i] = new Image(files[i]);
25
26         // 画像表示枠の準備します
27         ImageView[] ivs = new ImageView[3];
28         for(int i=0; i<ivs.length; i++)
29             ivs[i] = new ImageView();
30
31         // 画像表示枠に初期の画像を設定します
32         for(int i=0; i<ivs.length; i++)
33             ivs[i].setImage(imgs[i%imgs.length]);
34
35         // エフェクトを生成し、画像に適用します
36         PerspectiveTransform ptleft = new PerspectiveTransform();
37         ptleft.setUlx(160.0);
38         ptleft.setUly(50.0);
39         ptleft.setUrx(320.0);
40         ptleft.setUry(0.0);
41         ptleft.setLrx(320.0);
42         ptleft.setLry(240.0);
43         ptleft.setLlx(160.0);
44         ptleft.setLly(190.0);
45         PerspectiveTransform ptright = new PerspectiveTransform();
46         ptright.setUlx(0.0);
47         ptright.setUly(0.0);
48         ptright.setUrx(160.0);
49         ptright.setUry(50.0);
50         ptright.setLrx(160.0);
51         ptright.setLry(190.0);
52         ptright.setLlx(0.0);
53         ptright.setLly(240.0);
54
55         // エフェクトを両側の画像表示枠に適用します
56         ivs[0].setEffect(ptleft);
57         ivs[2].setEffect(ptright);
58
59         Thread th = new Thread(new SlideThread(imgs, ivs));
60         th.start();
61
62         // レイアウトHBoxを生成/設定します
63         HBox hb = new HBox();
64         ObservableList<Node> lst = hb.getChildren();
65         lst.addAll(ivs);
66
67         // シーンを生成/設定します
68         Scene scene = new Scene(hb);
69         scene.setFill(Color.BLACK);
70
71         // ステージを設定します
72         stage.setScene(scene);
73         stage.setTitle("★画像スライドショー★");
74
75         // ステージを表示します
76         stage.show();
77     }
78
79     public static void main(String[] args)
80     {
81         launch(args);
82     }
83 }
84
85 // スライドスレッド
86 class SlideThread extends Thread
87 {
88     // スライドを行う画像表示枠です
89     private Image[] imgs;
90     private ImageView[] ivs;
91
92     // コンストラクタ
93     public SlideThread(Image[] imgs, ImageView[] ivs)
94     {

```

```
95 // スライドさせる画像表示枠を受け取ります
96 this.imgs = imgs;
97 this.ivs = ivs;
98 }
99
100 // 1秒おきに画像をスライドします
101 public void run()
102 {
103     int count = ivs.length % imgs.length;
104     while(true)
105     {
106         try{
107             Thread.sleep(1000);
108         }catch(InterruptedException e){}
109
110         ivs[0].setImage(ivs[1].getImage());
111         ivs[1].setImage(ivs[2].getImage());
112         ivs[2].setImage(imgs[count]);
113         count++;
114         count %= imgs.length;
115     }
116 }
117 }
118
```