```java
import javafx.application.*;
import javafx.scene.*;
import javafx.scene.layout.*;
import javafx.scene.control.*;
import javafx.scene.paint.*;
import javafx.scene.image.*;
import javafx.scene.effect.*;
import javafx.scene.text.*;
import javafx.scene.input.*;
import javafx.stage.*;
import javafx.event.*;
import javafx.geometry.*;
import javafx.collections.*;

////////////////////////////////////////////////////////////
// 問１　マウスイベントが発生したら、以下のように種類に応じ…
////////////////////////////////////////////////////////////

public class Assignment7_1 extends Application
{
   public void start(Stage stage) throws Exception
   {
      // レイアウトVBoxを生成／設定します
      VBox vb = new VBox();

      // シーンを生成／設定します
      Scene scene = new Scene(vb);

      // イベントハンドラを設定します
      MyEventHandler mousehandler = new MyEventHandler();
      scene.addEventHandler(MouseEvent.ANY, mousehandler);

      // ステージを設定します
      stage.setScene(scene);
      stage.setTitle("マウスイベントの可視化");

      // ステージを表示します
      stage.show();
   }

   // イベントハンドラ（イベント処理）クラスの宣言
   private class MyEventHandler implements EventHandler<MouseEvent>
   {
      public void handle(MouseEvent e)
      {
         EventType<? extends MouseEvent> type = e.getEventType();
         if(type == MouseEvent.MOUSE_MOVED){
            System.out.print(">");
         }
         if(type ==  MouseEvent.MOUSE_PRESSED){
            System.out.print("P");
         }
         if(type ==  MouseEvent.MOUSE_RELEASED){
            System.out.print("R");
         }
         if(type ==  MouseEvent.MOUSE_CLICKED){
            System.out.println("C");
         }
         if(type ==  MouseEvent.MOUSE_DRAGGED){
            System.out.print("-");
         }
         if(type ==  MouseEvent.DRAG_DETECTED){
            System.out.print("!");
         }
      }
   }

   public static void main(String[] args)
   {
      launch(args);
   }
}

```

```java
import javafx.application.*;
import javafx.scene.*;
import javafx.scene.layout.*;
import javafx.scene.control.*;
import javafx.scene.paint.*;
import javafx.scene.image.*;
import javafx.scene.effect.*;
import javafx.scene.text.*;
import javafx.scene.input.*;
import javafx.stage.*;
import javafx.event.*;
import javafx.geometry.*;
import javafx.collections.*;

////////////////////////////////////////////////////////////
// 問２　写真閲覧アプリを作成しましょう。5枚の写真を読み込…
////////////////////////////////////////////////////////////

public class Assignment7_2 extends Application
{
  private Image[] img;    // 表示する画像の配列
  private int img_no;     // 現在表示している画像の添え字
  private ImageView iv;   // 画像表示クラス型の変数

  public void start(Stage stage) throws Exception
  {
    // 画像を準備します
    img = new Image[5];
    for(int i=0; i<img.length; i++)
      img[i] = new Image("photo"+i+".jpg");

    // 画像表示の準備をします
    iv = new ImageView();
    img_no = 0;
    iv.setImage(img[img_no]);

    // レイアウトVBoxを生成／設定します
    VBox vb = new VBox();
    ObservableList<Node> lst = vb.getChildren();
    lst.add(iv);

    // シーンを生成／設定します
    Scene scene = new Scene(vb);

    // イベントハンドラを設定します
    MyEventHandler mousehandler = new MyEventHandler();
    scene.addEventHandler(MouseEvent.ANY, mousehandler);

    // ステージを設定します
    stage.setScene(scene);
    stage.setTitle("画像閲覧アプリ");

    // ステージを表示します
    stage.show();
  }

  // イベントハンドラ（イベント処理）クラスの宣言
  private class MyEventHandler implements EventHandler<MouseEvent>
  {
    public void handle(MouseEvent e)
    {
      EventType<? extends MouseEvent> type = e.getEventType();
      // マウスが押される度に次の画像を表示します
      if(type ==  MouseEvent.MOUSE_PRESSED){
        img_no++;
        if(img_no >= img.length) img_no = 0;
        iv.setImage(img[img_no]);
      }
    }
  }

  public static void main(String[] args)
  {
    launch(args);
  }
}

```

```
import javafx.application.*;
import javafx.scene.*;
import javafx.scene.layout.*;
import javafx.scene.control.*;
import javafx.scene.paint.*;
import javafx.scene.image.*;
import javafx.scene.effect.*;
import javafx.scene.text.*;
import javafx.scene.input.*;
import javafx.stage.*;
import javafx.event.*;
import javafx.geometry.*;
import javafx.collections.*;

////////////////////////////////////////////////////////
// 問3　「パノラマ画像閲覧アプリ」を作成しましょう。横4000…
////////////////////////////////////////////////////////

public class Assignment7_3 extends Application
{
  private boolean slideLeft; // スライドの向き
  private ImageView iv;      // 画像ファイル
                             // - ハンドラから参照するため
  private boolean flag_stopThread;
                             // 画像のスライドスレッドの稼働フラグ

  public void start(Stage stage) throws Exception
  {
    // 画像を準備します
    Image im = new Image("pano1.jpg");

    // ビューポートを準備します
    // - 基となる画像の縦の長さを一辺とする正方形
    // - 初期値は画像の中央に設定
    Rectangle2D rt = new Rectangle2D(im.getWidth()/2+im.getHeight()/2, 0, im.getHeight(), im.getHeight());

    // 画像枠の準備をします
    iv = new ImageView(im);
    iv.setViewport(rt);

    // レイアウトVBoxを生成／設定します
    VBox vb = new VBox();
    ObservableList<Node> lst = vb.getChildren();
    lst.add(iv);

    // シーンを生成／設定します
    Scene scene = new Scene(vb);

    // イベントハンドラを設定します
    MyEventHandler mousehandler = new MyEventHandler();
    scene.addEventHandler(MouseEvent.ANY, mousehandler);

    // 画像をスライドするスレッドを開始します
                flag_stopThread = true;
    MyThread mt = new MyThread();
    mt.start();

    // イベントハンドラを設定します
    MyEventHandler2 mousehandler2 = new MyEventHandler2();
    stage.addEventHandler(WindowEvent.ANY, mousehandler2);

    // ステージを設定します
    stage.setScene(scene);
    stage.setTitle("パノラマ画像閲覧アプリ");
    stage.setResizable(false);
    // ウィンドウサイズを固定する場合に
    // 以下をしないと余白が勝手に出現します
    // バグです！そのうち修正されると思います
    stage.sizeToScene();

    // ステージを表示します
    stage.show();
  }

  // 画像のスライドを行うスレッドクラス
  private class MyThread extends Thread implements Runnable
  {
    private Rectangle2D rt;

    public void run(){
      // 現在のビューポートを取得
      rt = iv.getViewport();

      // ビューポートを左または右へ移動
      while(flag_stopThread){
        if(slideLeft){
          // 左へスライド
          if(rt.getMinX()-1 > 0)
            rt = new Rectangle2D(rt.getMinX()-1.5, 0, rt.getWidth(), rt.getHeight());
        }else{
          // 右へスライド
          if(rt.getMinX()+1 < iv.getImage().getWidth()-rt.getWidth())
```

```
93              rt = new Rectangle2D(rt.getMinX()+1.5, 0, rt.getWidth(), rt.getHeight());
94          }
95          iv.setViewport(rt);
96
97          try{
98            sleep(5);
99          }catch(Exception e){}
100       }
101     }
102   }
103
104   // イベントハンドラ（イベント処理）クラスの宣言
105   private class MyEventHandler implements EventHandler<MouseEvent>
106   {
107     public void handle(MouseEvent e)
108     {
109       EventType<? extends MouseEvent> type = e.getEventType();
110       // マウスが動く度に次の画像を表示します
111       if(type ==  MouseEvent.MOUSE_MOVED){
112         double mx = e.getX();
113         Rectangle2D rt = iv.getViewport();
114
115         // マウスが画像の半分より左にあるか右にあるか判断します
116         // - 左にあると、画像を左へスライド
117         // - 右にあると、画像を右へスライド
118         if(mx > rt.getWidth()/2)
119           slideLeft = false;
120         else
121           slideLeft = true;
122       }
123     }
124         }
125
126   // イベントハンドラ（イベント処理）クラスの宣言
127   private class MyEventHandler2 implements EventHandler<WindowEvent>
128   {
129     public void handle(WindowEvent e)
130     {
131       EventType<WindowEvent> type = e.getEventType();
132       // ウィンドウが閉じられようとするとスレッドを停止
133       if(type ==  WindowEvent.WINDOW_CLOSE_REQUEST){
134         flag_stopThread = false;
135       }
136     }
137   }
138
139   public static void main(String[] args)
140   {
141     launch(args);
142   }
143 }
144
```