Санкт-Петербургский государственный университет информационных технологий,
механики и оптики

Кафедра «Компьютерные технологии»

В.О. Дронь, А.А. Плодовитова

СИСТЕМА УПРАВЛЕНИЯ МОДЕЛЬЮ ФОТОАППАРАТА

Программирование с явным выделением состояний

ПРОЕКТНАЯ ДОКУМЕНТАЦИЯ

Проект создан в рамках
«Движения за открытую проектную документацию»
http://is.ifmo.ru

Санкт-Петербург
2004

Оглавление

1. Введение

В настоящее время в мире существует огромное количество цифровых устройств, для управления которыми необходимо специальное программное обеспечение. В рамках данной работы авторы хотели бы показать, насколько может быть удобным для реализации этих целей применение *switch*-технологии [1,2].

Основным этапом создания программы в этом случае становится проектирование. На этом этапе строятся управляющие автоматы и определяется их взаимодействие. Непосредственное написание кода существенно упрощается, поскольку он становится «изоморфен» графам переходов используемых автоматов. Также облегчается понимание проекта и упрощается внесение в него изменений.

В данной работе с использованием switch-технологии реализована система управления моделью цифрового фотоаппарата.

Программа написана на языке *Java* [3].

2. Постановка задачи

Целью данной работы является демонстрация принципов автоматного программирования на примере создания модели фотоаппарата.

В модели реализованы два основных режима:

- просмотр отснятых кадров.
- автосъемка;

На рис.1, 2 приведен внешний вид модели.

Рис.1. Внешний вид модели (режим просмотра отснятых кадров)

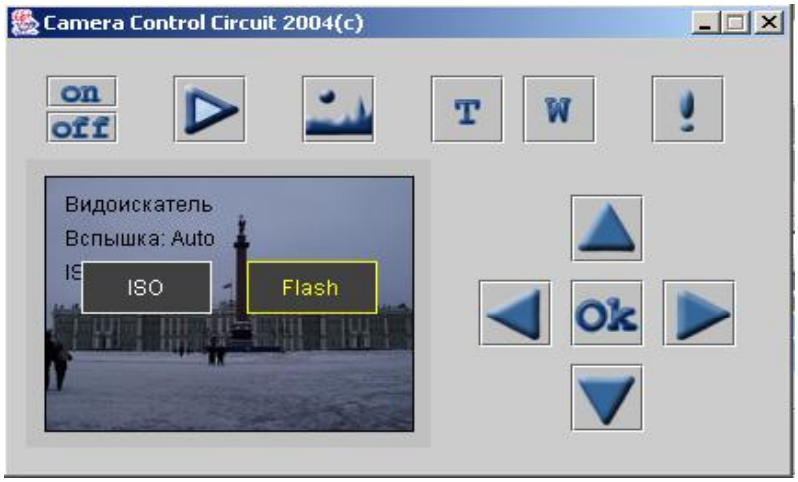

Рис.2. Внешний вид модели (режим автосъемки)

Опишем структуру панели управления. В верхней ее части расположены кнопки:

- «Включение» (*on*);
- «Выключение» (*off*);
- «Просмотр отснятых кадров» (►);
- «Автосъемка» (изображен пейзаж);
- «Уменьшение изображения» (*T* от слова «Tighter» - уже);
- «Увеличение изображения» (*W* от слова «Wider» - шире);
- «Съемка».

В нижней части панели находятся:

- экран, выполняющий функции видоискателя и позволяющий просматривать отснятые кадры;
- кнопки «Вправо», «Влево», «Вверх», «Вниз», позволяющие двигать изображение и переключаться между пунктами меню;
- кнопка «Меню» (*Ok*), функциональность которой описана ниже.

При просмотре отснятых кадров пользователь имеет возможность увеличивать размер изображений с помощью соответствующей кнопки и просматривать его кнопками «Вправо», «Влево», «Вверх», «Вниз». При нажатии кнопки «Уменьшение …» изображение возвращается к стандартному размеру.

В режиме съемки пользователь также может приближать или удалять объект съемки соответствующими кнопками. При нажатии кнопки «Меню» на экране видоискателя вызывается меню, состоящее из пунктов:

- «Установка светочувствительности» (*ISO*);
- «Выбор режима вспышки» (*Flash*).

Вспышка может быть включена, выключена и находиться в режиме «Автовспышка», что обеспечивается с помощью вложенного меню.

Для светочувствительности могут быть установлены следующие значения: 100, 200 и 400.

При нажатии кнопки «Съемка» происходит фотографирование. Изображение, которое в этот момент будет находиться в окне видоискателя, добавится к отснятым кадрам. Качество и вид изображения будут зависеть от установленных значений параметров вспышки и светочувствительности.

3. Структура программы

Диаграмма классов программы приведена на рис. 3.

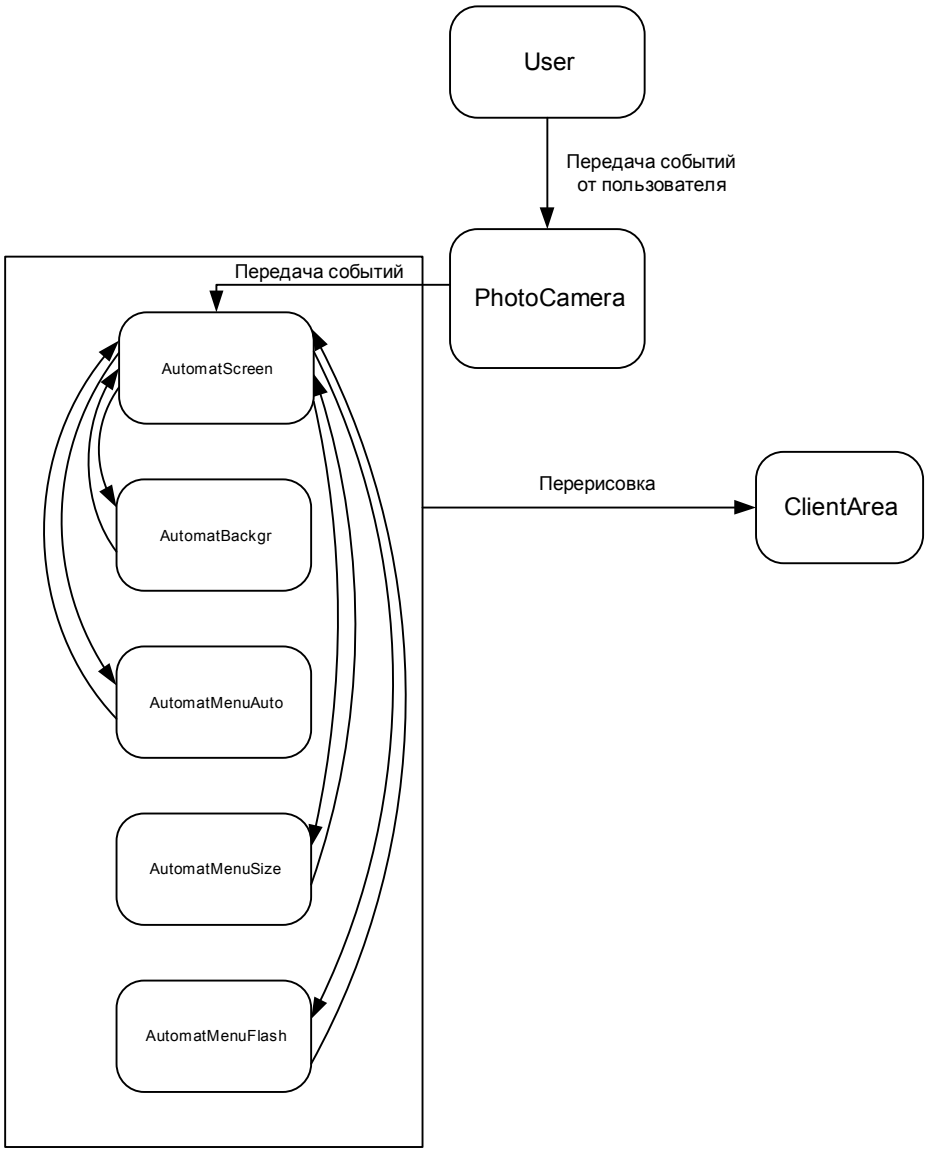

Рис.3. Диаграмма классов

Как следует из рисунка, классы могут быть разделены на две разновидности: вспомогательные и автоматные.

Перечислим вспомогательные классы:

- *PhotoCamera* (создает окно программы, инициализирует все автоматы и другие классы);
- *ClientArea* (рисует основную клиентскую область);
- *TopPanel* (отрисовывает верхний ряд кнопок);
- *ControlPanel* (отрисовывает навигационные кнопки правого меню);
- *ImageMap* (обеспечивает работу с графическими файлами);
- *Log* (отвечает за поддержку логфайла).

При построении автоматных классов используется наследование. Класс «Automat» является базовым для всех остальных автоматов и содержит их общие поля и методы:

- *Y0* (номер состояния автомата);
- *GetState* (возвращает номер состояния автомата);
- *SetState* (устанавливает номер состояния автомата);
- *Name* (имя автомата);
- *GetName* (возвращает имя автомата);
- *SetName* (устанавливает номер состояния автомата);
- *AutomatType* (тип автомата: A0, A1, …);
- *GetType* (возвращает тип автомата);
- *SetType* (устанавливает тип автомата).

Программа содержит пять классов, наследуемых от базового:

- «AutomatScreen» (A0);
- «AutomatBackGr» (A1);
- «AutomatMenuAuto» (A2);
- «AutomatMenuSize» (A3);
- «AutomatMenuFlash» (A4).

Схема наследования приведена на рис. 4.

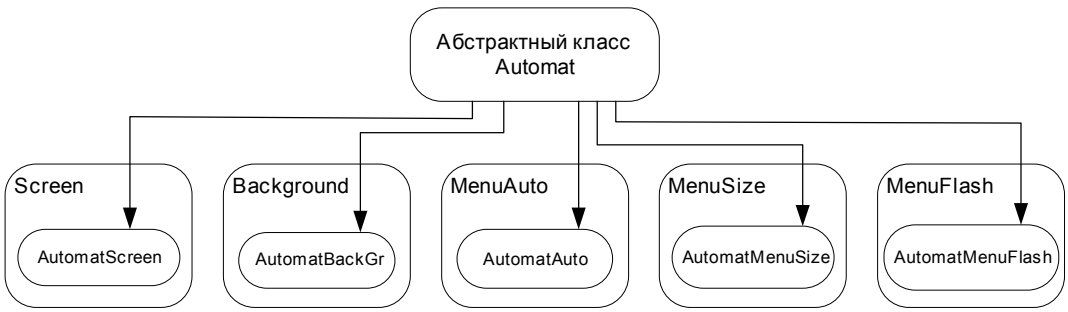

Рис.4. Схема наследования

Схема взаимодействия автоматов приведена на рис. 5.

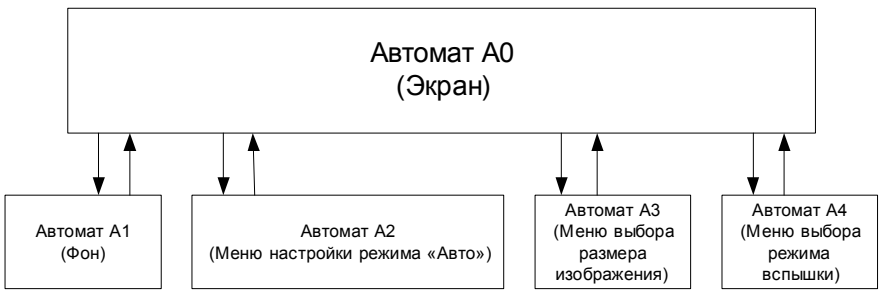

Рис.5. Схема взаимодействия автоматов

4. Класс *Screen*

4.1. Описание класса

Класс *Screen* предназначен для реализации экрана, управляемого автоматом *AutomatScreen*.

4.2. Автомат AutomatScreen (A0)

4.2.1. Описание событий

e1 – Нажатие кнопки «Вверх».
e2 – Нажатие кнопки «Вниз».
e3 – Нажатие кнопки «Вправо».
e4 – Нажатие кнопки «Влево».
e7 – Нажатие кнопки «Увеличить изображение».
e8 – Нажатие кнопки «Уменьшить изображение».
e9 – Нажатие кнопки «Автосъемка».
e10 – Нажатие кнопки «Просмотр отснятых кадров».
e11 – Нажатие кнопки «Меню».
e11_0 – Включение меню настройки параметров автоматической съемки.
Поступает от автомата A1.
e11_1 – Отказ от какого-либо меню. Поступает от автоматов A2, A3 или A4.
e12 – Выбор меню настройки размеров изображения. Поступает от автомата A2.
e13 – Выбор меню настройки параметров вспышки. Поступает от автомата A2.
e14 – Выключение фотоаппарата.
e15 – Включение фотоаппарата.
e16 – Нажатие кнопки «Съемка».
e17 – Завершение съемки. Поступает от таймера.

4.2.2. Описание входных переменных

x1 – Установлен режим «Автоматическая съемка».
x2 – Установлен режим «Просмотр кадров».

4.2.3. Описание выходных воздействий

z0_01 – Послать соответствующее сообщение автомату A1.
z0_1 – Изменить входные переменные, перерисовать экран.
z0_11 – Послать соответствующее сообщение A1.
z0_2 – Перерисовать экран.
z0_21 – Послать соответствующее сообщение A1.
z0_22 – Послать соответствующее сообщение A2.
z0_3 – Перерисовать экран.
z0_31 – Послать соответствующее сообщение A.
z0_32 – Послать соответствующее сообщение A3.
z0_4 – Перерисовать экран.
z0_41 – Послать соответствующее сообщение A1.
z0_42 – Послать соответствующее сообщение A4.
z0_5 – Перерисовать экран.
z0_6 – Изменить значения входных переменных, перерисовать экран.
z0_61 – Перерисовать экран.

4.2.4. Схема связей автомата

Схема связей автомата A0 приведена на рис. 6.

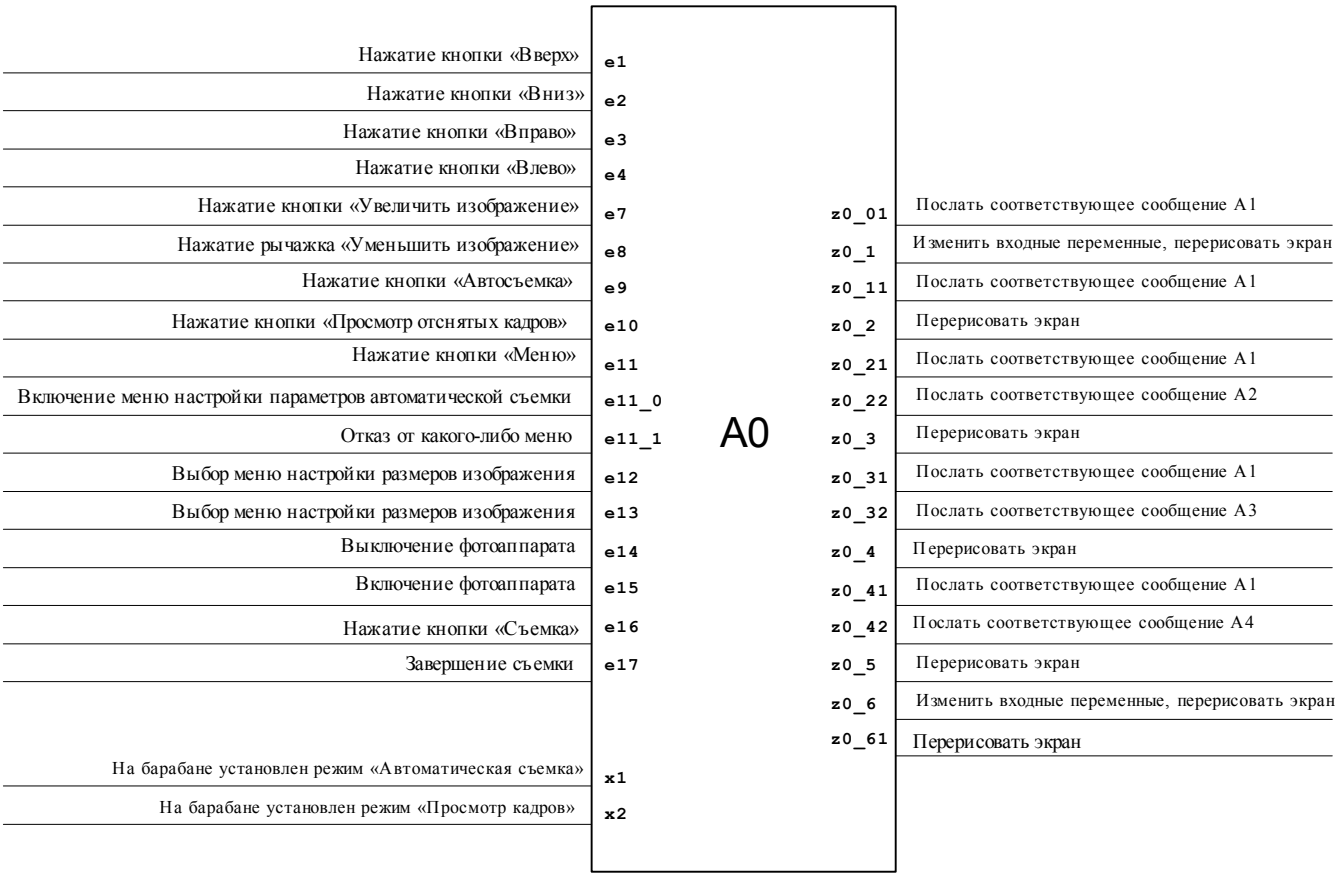

Рис. 6. Схема связей автоматов

4.2.5. Граф переходов

Граф переходов автомата «AutomatScreen» изображен на рис. 7.

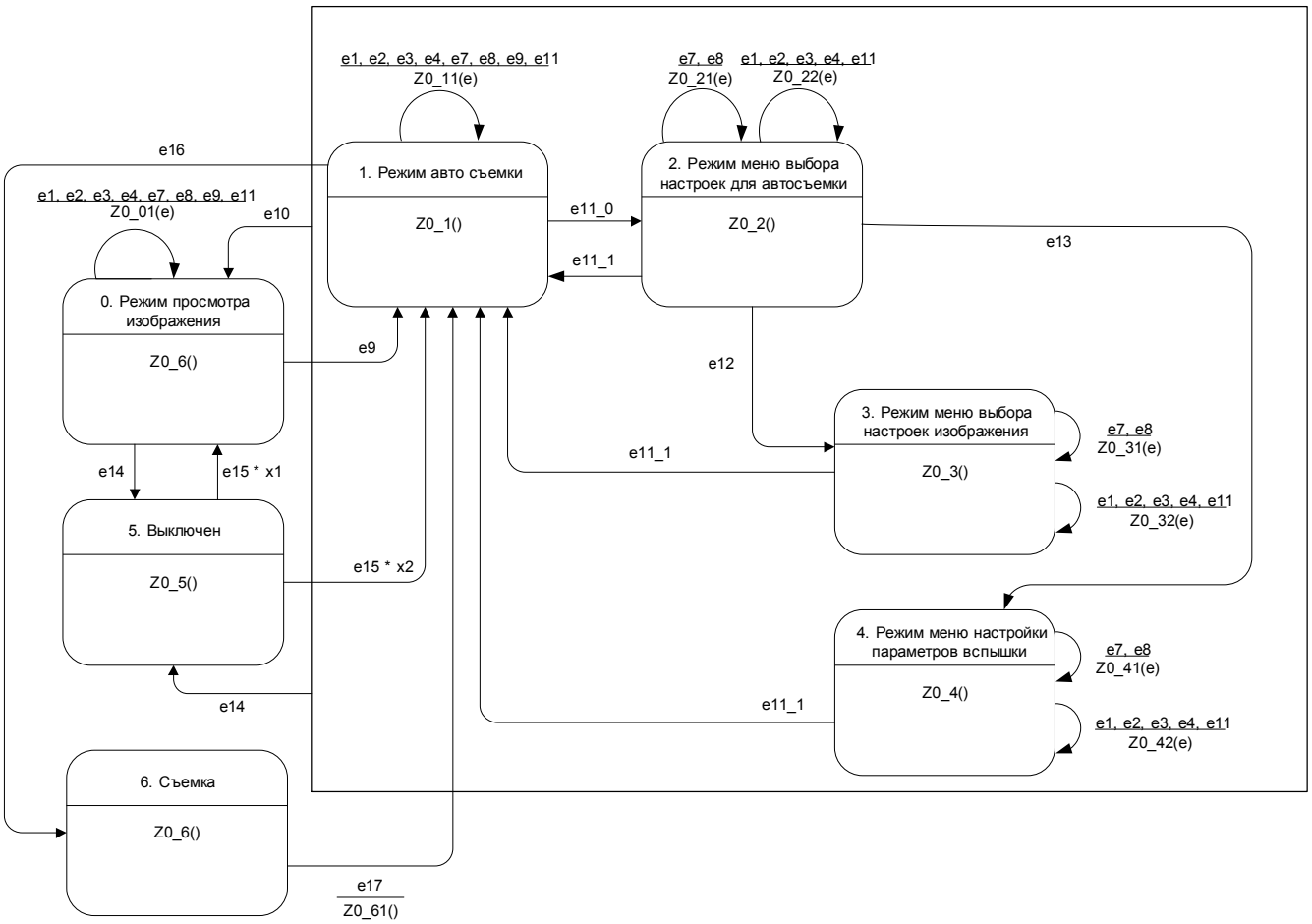

Рис.7. Граф переходов автомата «Экран»

5. Класс *BackGr*

5.1. Описание класса

Данный класс отвечает за перерисовку экрана видоискателя.

5.2. Автомат AutomatBackG (A1)

5.2.1. Описание событий

e1 – Нажатие кнопки «Вверх». Поступает от автомата A0 (Экран)
e2 – Нажатие кнопки «Вниз». Поступает от автомата A0 (Экран)
e3 – Нажатие кнопки «Вправо». Поступает от автомата A0 (Экран)
e4 – Нажатие кнопки «Влево». Поступает от автомата A0 (Экран)
e7 – Нажатие рычажка «Увеличить изображение». Поступает от автомата A0 (Экран)

e8 – Нажатие рычажка «Уменьшить изображение». Поступает от автомата A0 (Экран)
e9 – Включение режима «Автоматическая съемка». Поступает от автомата A0 (Экран)
e10 – Включение режима «Просмотр кадров». Поступает от автомата A0 (Экран)
e11 – Нажатие кнопки «Меню». Поступает от автомата A0 (Экран)

5.2.2. Описание выходных воздействий

z1_0 – Перерисовать экран.
z1_01 – Увеличить параметр «Номер просматриваемого кадра». Перерисовать экран.
z1_02 – Уменьшить параметр «Номер просматриваемого кадра». Перерисовать экран.
z1_1 – Перерисовать экран.
z1_11 – Увеличить параметр «Номер просматриваемого кадра». Перерисовать экран.
z1_12 – Уменьшить параметр «Номер просматриваемого кадра». Перерисовать экран.
z1_2 – Установить начальные параметры скроллинга (например, вертикальный скроллинг - 0, горизонтальный скроллинг – 0). Перерисовать экран.
z1_21 – Увеличить изображение. Перерисовать экран.
z1_22 – Увеличить параметр горизонтального скроллинга. Перерисовать экран.
z1_23 – Уменьшить параметр горизонтального скроллинга. Перерисовать экран.
z1_24 – Увеличит параметр вертикального скроллинга. Перерисовать экран.
z1_25 – Уменьшить параметр вертикального скроллинга. Перерисовать экран.
z1_3 – Перерисовать экран.
z1_31 – Увеличить изображение. Перерисовать экран.
z1_32 – Уменьшить изображение. Перерисовать экран.
z1_33 – Послать автомату A0 сообщение e11_0.

5.2.3. Схема связей автомата

Схема связей автомата A1 приведена на рис. 8.

Рис. 8. Схема связей автомата A1

5.2.4. Граф переходов

Граф переходов автомата «Фон» изображен на рис. 9.

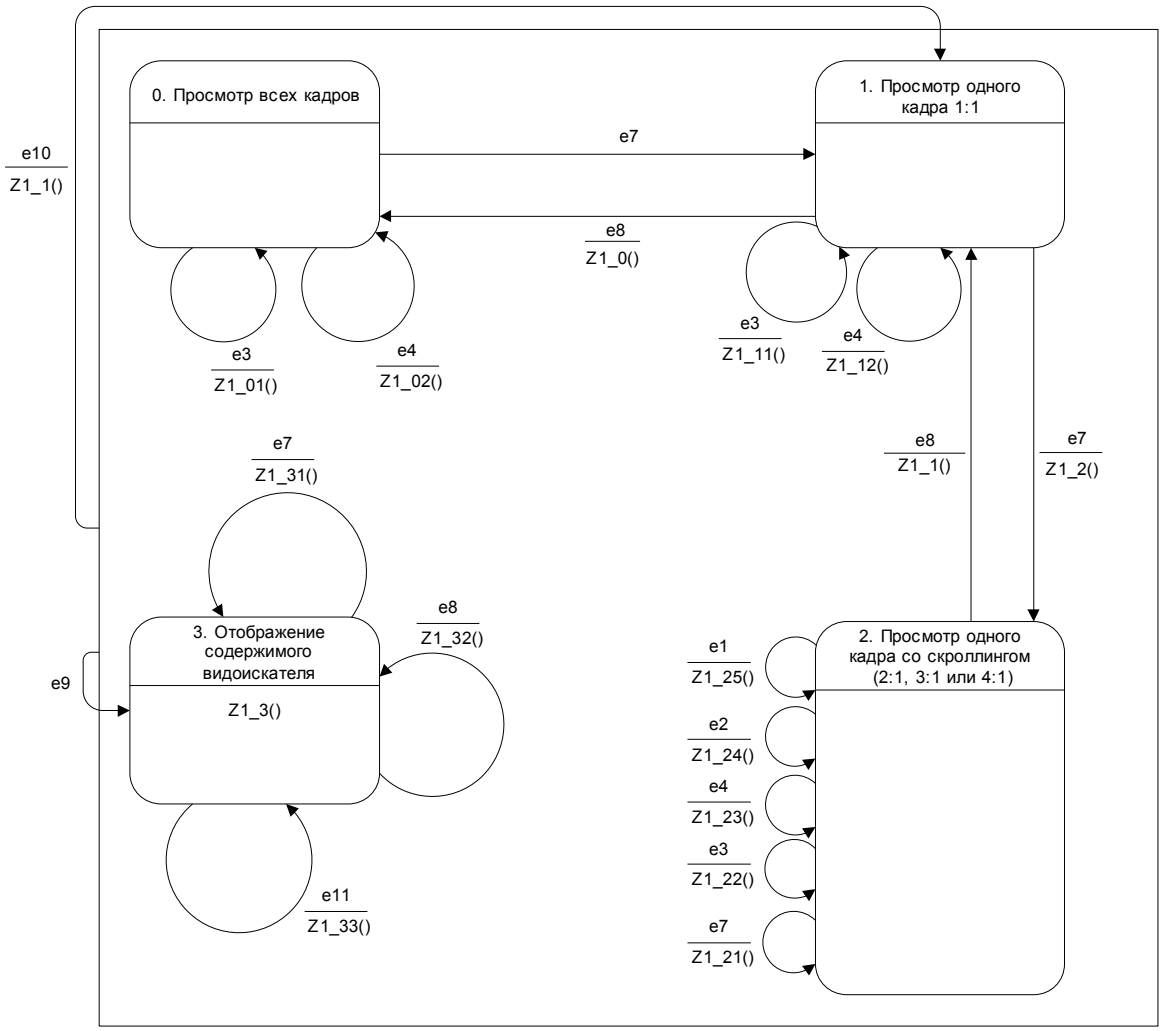

Рис. 9. Граф переходов автомата «Фон»

6. Класс *MenuAuto*

6.1. Описание класса

Класс *MenuAuto* отвечает за режим съемки.

6.2. Автомат AutomatMenuAuto (A2)

6.2.1. Описание событий

e3 – Нажатие кнопки «Вправо». Поступает от автомата A0 (Экран).
e4 – Нажатие кнопки «Влево». Поступает от автомата A0 (Экран).
e11 – Нажатие кнопки «Меню». Поступает от автомата A0 (Экран)

6.2.2. Входная переменная

x1 – Меню отображается на экране

6.2.3. Описание выходных воздействий

z2_0 – Послать автомату A0 сообщение e11_1. Обнулить значение входной переменной x1.
z2_01 – Установить единицу во входную переменную x1.
z2_11 – Послать автомату A0 сообщение e12. Обнулить значение входной переменной x1.
z2_12 – Послать автомату A0 сообщение e13. Обнулить значение входной переменной x1.

6.2.4. Схема связей автомата

Схема связей автомата A2 приведена на рис.10.

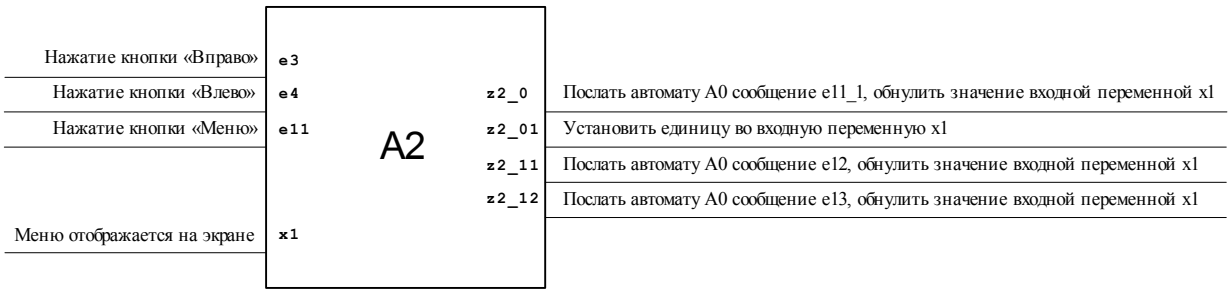

Рис. 10. Схема связей автомата A2

6.2.5. Граф переходов

Граф переходов автомата «Меню настройки режима «Авто» изображен на рис. 11.

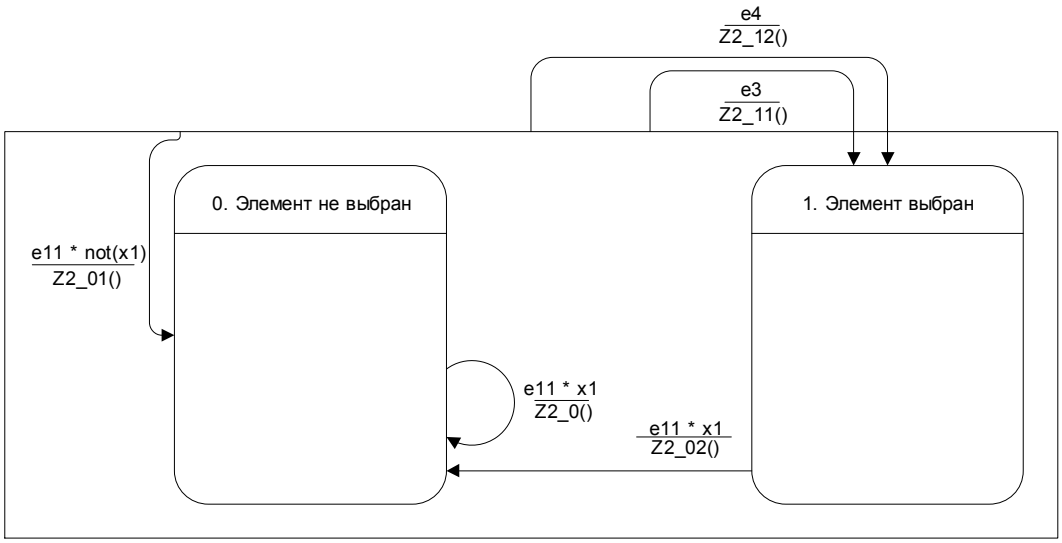

Рис. 11. Граф переходов автомата «Меню настройки режима «Авто»

7. Класс *MenuSize*

7.1. Описание класса

Класс «MenuSize» отвечает за выбор размера изображения.

7.2. Автомат *AutomatMenuSize (A3)*

7.2.1. Описание событий

e1 – Нажатие кнопки «Вверх». Поступает от автомата A0 (Экран).
e2 – Нажатие кнопки «Вниз». Поступает от автомата A0 (Экран).
e4 – Нажатие кнопки «Влево». Поступает от автомата A0 (Экран).
e11 – Нажатие кнопки «Меню». Поступает от автомата A0 (Экран).

7.2.2. Описание входных переменных

x1 – Меню отображается на экране.

7.2.3. Описание выходных воздействий

z3_0 – Послать автомату A0 сообщение e11_1. Обнулить значение входной переменной x1.
z3_01 – Установить единицу во входную переменную x1.
z3_11 – Изменить значение параметра «Номер выбранного элемента».
z3_21 – Изменить значение параметра «Номер выбранного элемента».
z3_31 – Изменить значение параметра «Номер выбранного элемента».

7.2.4. Схема связей автомата

Схема связей автомата A3 приведена на рис. 12.

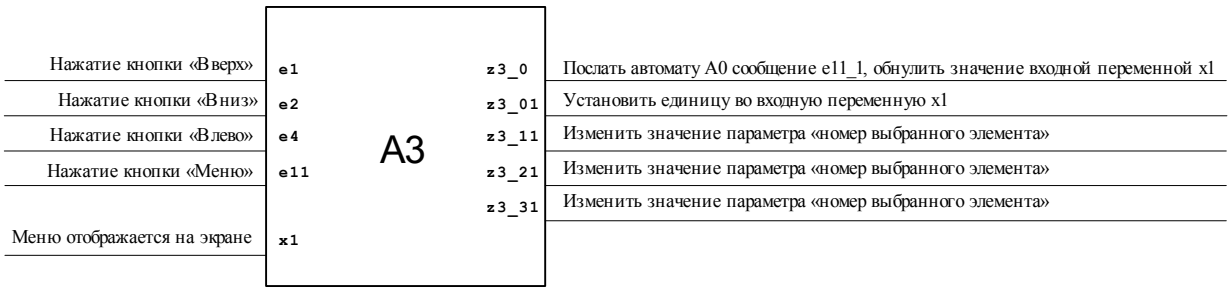

Рис. 12. Схема связей автомата A3

7.2.5. Граф переходов

Граф переходов автомата «Меню выбора режима вспышки» изображен на рис. 13.

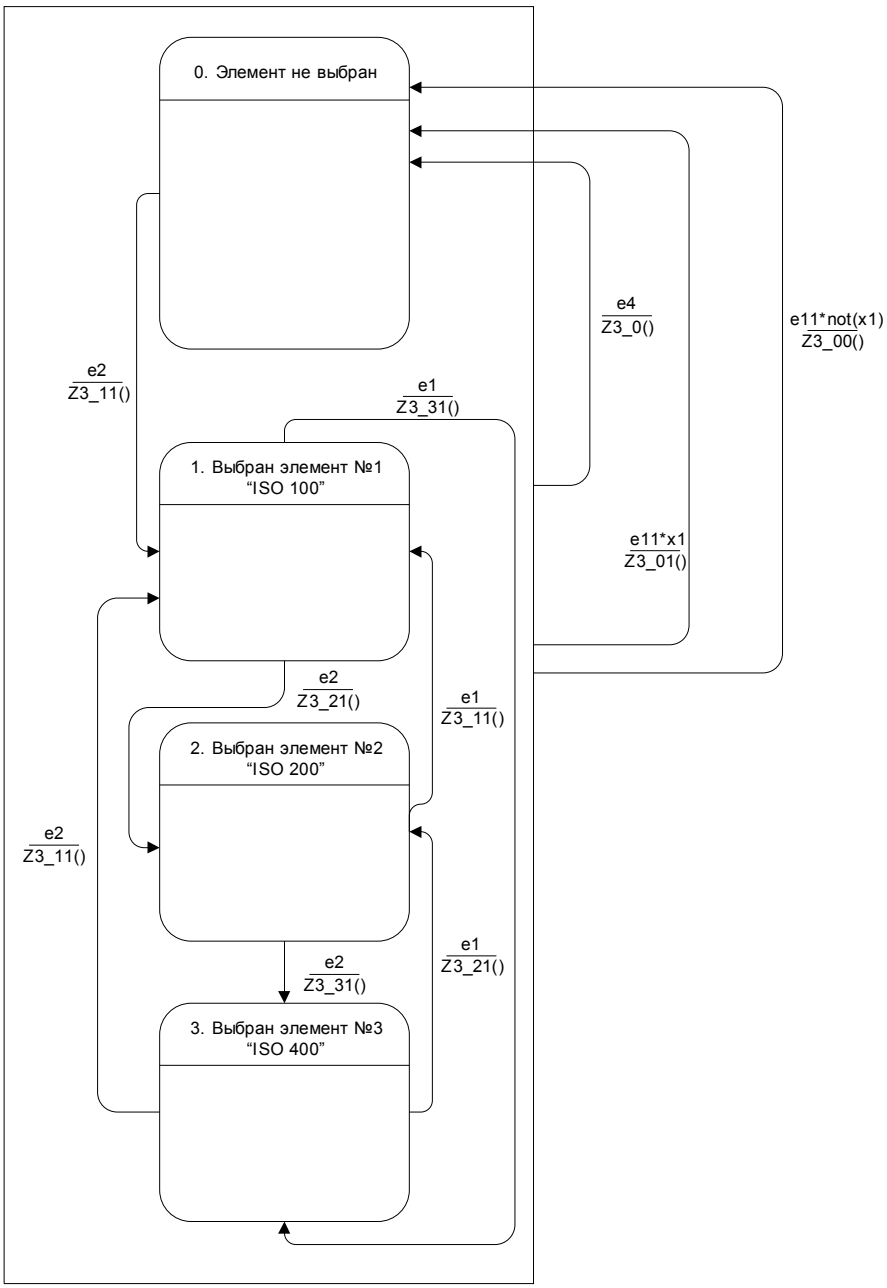

Рис. 13. Граф переходов автомата «Меню выбора режима вспышки»

8. Класс *MenuFlash*

8.1. Описание класса

Класс *MenuFlash* отвечает за выбор режима вспышки.

8.2. Автомат AutomatMenuFlash (A4)

8.2.1. Описание событий

e1 – Нажатие кнопки «Вверх». Поступает от автомата A0 (Экран)
e2 – Нажатие кнопки «Вниз». Поступает от автомата A0 (Экран)
e4 – Нажатие кнопки «Влево». Поступает от автомата A0 (Экран)
e11 – Нажатие кнопки «Меню». Поступает от автомата A0 (Экран)

8.2.2. Описание входных переменных

x1 – Меню отображается на экране

8.2.3. Описание выходных воздействий

z3_0 – Послать автомату A0 сообщение e11_1. Обнулить значение входной переменной x1.
z3_01 – Установить единицу во входную переменную x1.
z3_11 – Изменить значение параметра «Номер выбранного элемента».
z3_21 – Изменить значение параметра «Номер выбранного элемента».
z3_31 – Изменить значение параметра «Номер выбранного элемента».

8.2.4. Схема связей автомата

Схема связей автомата A4 приведена на рис. 14.

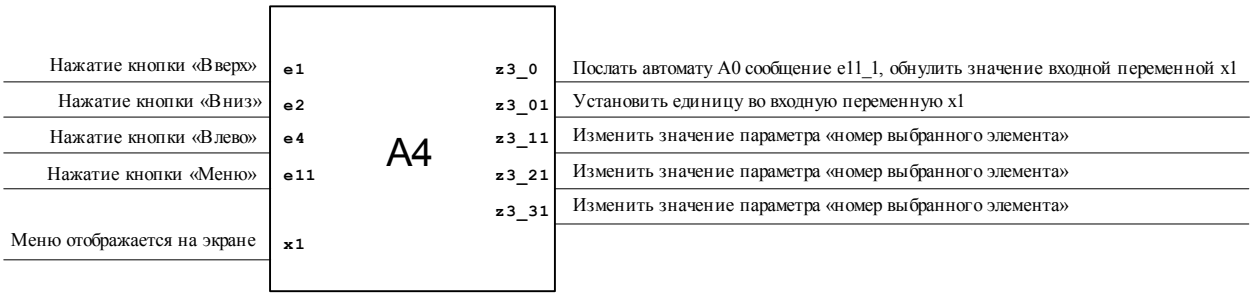

Рис. 14. Схема связей автомата A4

8.2.5. Граф переходов

Граф переходов автомата «Меню выбора режима вспышки» изображен на рис. 15.

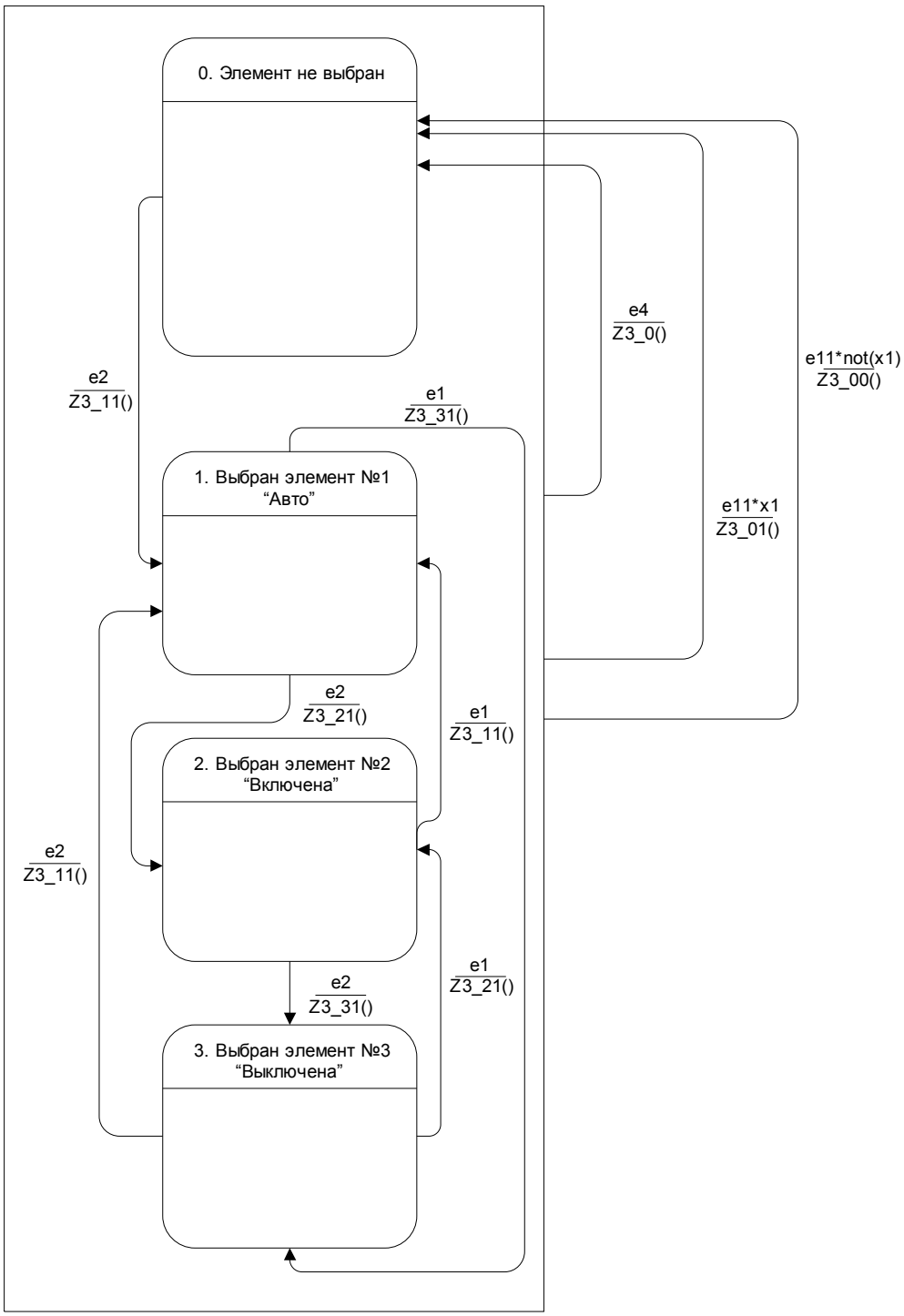

Рис. 15. Граф переходов автомата «Меню выбора режима вспышки»

Заключение

Данный проект показывает, что автоматный подход в программировании может быть удобным для решения целого ряда задач. Особенно упрощается процесс изменения программы, так как код, изоморфный графам переходов, удобен для чтения и внесения изменений.

Литература

1. *Шалыто А.А.* SWITCH-технология. Алгоритмизация и программирование задач логического управления. СПб.: Наука. 1998.
2. *Шалыто А.А., Туккель Н.И.* SWITCH-технология – автоматный подход к созданию программного обеспечения «реактивных» систем //Программирование. 2001. № 5. http://is.ifmo.ru. Раздел «Статьи».
3. *Ноутон П., Шилдт Л.* Java 2. СПб.: БХВ-Петербург, 2001.

Приложение. Текст программы

PhotoCamera. java

```
import java.net.*;
import javax.swing.*;
import java.beans.*;
import java.awt.*;
import java.awt.event.*;
import java.awt.Color.*;
import java.util.*;
import java.text.MessageFormat;

/**
* <code>PhotoCamera</code> - это класс, описывающий приложение.
*/
public final class PhotoCamera extends JFrame
implements ActionListener, MouseMotionListener, MouseListener, ComponentListener{

     public Screen screen;
     public BackGr backGr;
     public MenuAuto menuAuto;
     public MenuChooseSize menuChooseSize;
     public MenuChooseFlashMode menuChooseFlashMode;

     public ClientArea clientArea = new ClientArea(this);
     public ControlPanel controlPanel = new ControlPanel();
     public TopPanel topPanel = new TopPanel();

     public static void main(String[] args) {
         Frame f;
         f = new PhotoCamera("Camera Control Circuit 2004(c)");
         f.setResizable(false);
     };

     public void paintClient(Graphics g){
         screen.draw(g);
     };

     public PhotoCamera(String s) {
         super(s);
         int i;

         screen = new Screen(this);
         backGr = new BackGr(screen);
         menuAuto = new MenuAuto(screen);

         menuChooseSize = new MenuChooseSize(screen);
         menuChooseFlashMode = new MenuChooseFlashMode(screen);
         //getContentPane().setLayout(new BorderLayout());
         getContentPane().setLayout(null);

         clientArea.setBounds(10, 70, 220, 170);
         getContentPane().add(clientArea);

         controlPanel.addActionListener(this);
//         controlPanel.setBounds(250, 10, 400, 250);
         getContentPane().add(controlPanel);//, BorderLayout.WEST);

         topPanel.addActionListener(this);
         topPanel.setBounds(0, 0, 430, 70);
         getContentPane().add(topPanel);//, BorderLayout.WEST);
//         clientArea.setBackground(Color.black);
```

```
      clientArea.addMouseMotionListener(this);
      clientArea.addMouseListener(this);

      addComponentListener(this);
      addWindowListener(new WindowAdapter() {
          public void windowClosing(WindowEvent e){
              System.exit(0);
          }
      });

      setBounds(100, 100, 400, 200);
      setSize(430, 280);
      setVisible(true);
      repaint();
  }

public void actionPerformed(ActionEvent e)     {
    Object source = e.getSource();
    if (source == controlPanel.up) {

        Log.writeln("Press button Up");
        screen.pressUp();

    }else if(source == controlPanel.down) {

        Log.writeln("Press button Down");
        screen.pressDown();

    }else if(source == controlPanel.right) {

        Log.writeln("Press button Right");
        screen.pressRight();

    }else if(source == controlPanel.left) {

        Log.writeln("Press button Left");
        screen.pressLeft();

    }else if(source == topPanel.zoomIn) {

        Log.writeln("Press button ZoomIn");
        screen.pressZoomIn();

    }else if(source == topPanel.zoomOut) {

        Log.writeln("Press button ZoomOut");
        screen.pressZoomOut();

    }else if(source == topPanel.autoMode) {

        Log.writeln("Press button AutoMode");
        screen.pressAutoMode();

    }else if(source == topPanel.viewMode) {

        Log.writeln("Press button ViewMode");
        screen.pressSlideView();

    }else if(source == controlPanel.menubut) {

        Log.writeln("Press button MenuBut");
        screen.pressMenubut();

    }else if(source == topPanel.powerOn) {
```

```
            Log.writeln("Press button PowerOn");
            screen.pressOn();

        }else if(source == topPanel.powerOff) {

            Log.writeln("Press button PowerOff");
            screen.pressOff();

        }else if(source == topPanel.shoot) {

            Log.writeln("Press button shoot");
            screen.shoot();

        };

        clientArea.repaint();
    };

    public void mouseDragged(MouseEvent e) {
    };

    public void mouseMoved(MouseEvent e) {
    };

    public void mouseClicked(MouseEvent e) {
    };


    public void mouseEntered(MouseEvent e) {}


    public void mouseExited(MouseEvent e) {}
    public void mousePressed(MouseEvent e) {

    }

    public void mouseReleased(MouseEvent e) {}

    public void componentResized(ComponentEvent e) {
    }

    public void componentMoved(ComponentEvent e) {}
    public void componentShown(ComponentEvent e) {}
    public void componentHidden(ComponentEvent e) {}
}
```

Screen. java

```
import javax.swing.*;
import java.awt.*;
import java.util.*;
import java.util.Timer;
import java.lang.reflect.Method;
import java.io.*;

/**
 * класс "Экран"
 */
public class Screen {

    /**
     * Класс "Автомат для управления экраном"
     */
    class AutomatScreen extends Automat {
```

```
/**
 * Конструктор
 */
public AutomatScreen() {
    setState(5);
    setName("Automat A0");
};


/**
 * Функция-флажок. Просто говорит, верно ли, что  текущий режим - автосъемка
 */
public boolean x1() {
    return (mode == 0);
};

/**
 * Функция-флажок. Просто говорит, верно ли, что  текущий режим - slideshow
 */
public boolean x2() {
    return (mode == 1);
};

/**
 * Функции-выходы
 */
public void z0_1() {
    mode = 0; //режим автосъемки
    onoff = true;
};

public void z0_2() {
    owner.menuAuto.pressMenubut();
};

public void z0_3() {
    //owner.menuChooseSize.A3();
    owner.menuChooseSize.pressMenubut();
};

public void z0_4() {
    owner.menuChooseFlashMode.pressMenubut();
};

public void z0_5() {
    onoff = false;
};

public void z0_6() {
    mode = 1;
    onoff = true;
};

public void z0_61() {
    mode = 0;
    shootTimerTask.cancel();
    onoff = true;

    int slideTotal = owner.backGr.getSlideTotal() + 1;
    owner.backGr.setSlideTotal(slideTotal);
    String fileOutName = String.valueOf(slideTotal) + "img.gif";

    String fileInName;
    int zoom = owner.backGr.getZoom();
    if (zoom == 1) {
```

```
                fileInName = "sight4" + owner.menuChooseSize.getSelectedItem();
                if (owner.menuChooseFlashMode.getSelectedItemIndex() == 1) {
                    fileInName = fileInName + "flashOn";
                }

            }else{
                fileInName = "sight4zoom" + zoom;
            }

            fileInName = fileInName + ".gif";

            File fileIn = new File(System.getProperty("user.dir") + "/img/sight/" +
fileInName);
            File fileOut = new File(System.getProperty("user.dir") + "/img/" + fileOutName);
            try {
                BufferedInputStream reader = new BufferedInputStream(new
FileInputStream(fileIn));
                //FileInputStream reader = new FileInputStream(fileIn);
                BufferedOutputStream stream = new BufferedOutputStream(new
FileOutputStream(fileOut));
                int n;
                do{
                    n = reader.read();
                    if (n != -1) {
                        stream.write(n);
                    };

                }while (n != -1);
                stream.close();
            }catch(java.io.FileNotFoundException exp) {
            }catch(java.io.IOException exp) {
            };

        };

        public void z0_7() {
            mode = 2;
            onoff = true;
            shootTimerTask = new ShootTimerTask();
            timer.schedule(shootTimerTask, 1000, 1000);
        };

        /**
         * Непосредственно функция автомата
         */
        public void A(int e) {
            int y0 = getState();
            int yold = y0;

            Method method = null;
            Vector methodParametr = new Vector();
            Object object = null;

            BackGr backGr = owner.backGr;

            Class classes[] = new Class[1];
            classes[0] = Integer.class;

            try {
                switch (y0) {
                    case 0:

                        if (e == 14) {
                            y0 = 5;
                        }
```

```
            if (e == 110) {
                y0 = 2;
            }
            if (e == 7) {
                y0 = 0;
                method = BackGr.AutomatBackGr.class.getMethod("A", classes);
                methodParametr.add(new Integer(7));
                object = backGr.A1;
            }

            if (e == 8) {
                y0 = 0;
                owner.backGr.A1.A(8);
            }
            if (e == 1) {
                y0 = 0;
                owner.backGr.A1.A(1);
            }
            if (e == 2) {
                y0 = 0;
                owner.backGr.A1.A(2);
            }
            if (e == 3) {
                y0 = 0;
                owner.backGr.A1.A(3);
            }
            if (e == 4) {
                y0 = 0;
                owner.backGr.A1.A(4);
            }
            if (e == 9) {
                y0 = 1;
                owner.backGr.A1.A(9);
            }
            if (e == 10) {
                y0 = 0;
                owner.backGr.A1.A(10);
            }
            if (e == 11) {
                y0 = 0;
                method = BackGr.AutomatBackGr.class.getMethod("A", classes);
                methodParametr.add(new Integer(11));
                object = backGr.A1;
            }
            break;

        case 1:
            if (e == 10) {
                y0 = 0;
                owner.backGr.A1.A(10);
            }
            if (e == 14) {
                y0 = 5;
            }
            if (e == 110) {
                y0 = 2;
            }
            if (e == 7) {
                y0 = 1;
                owner.backGr.A1.A(7);
            }
            if (e == 8) {
                y0 = 1;
                owner.backGr.A1.A(8);
            }
            if (e == 1) {
```

```
            y0 = 1;
            owner.backGr.A1.A(1);
        }
        if (e == 2) {
            y0 = 1;
            owner.backGr.A1.A(2);
        }
        if (e == 3) {
            y0 = 1;
            owner.backGr.A1.A(3);
        }
        if (e == 4) {
            y0 = 1;
            owner.backGr.A1.A(4);
        }
        if (e == 9) {
            y0 = 1;
            owner.backGr.A1.A(9);
        }
        if (e == 11) {
            y0 = 1;
            method = BackGr.AutomatBackGr.class.getMethod("A", classes);
            methodParametr.add(new Integer(11));
            object = backGr.A1;
        }
        if (e == 16) {
            y0 = 6;
        }
        break;

    case 2:
        if (e == 3) {
            y0 = 2;
            method = MenuAuto.AutomatMenuAuto.class.getMethod("A", classes);
            methodParametr.add(new Integer(3));
            object = owner.menuAuto.A2;
            //owner.menuAuto.pressRight();
        }
        if (e == 4) {
            y0 = 2;
            method = MenuAuto.AutomatMenuAuto.class.getMethod("A", classes);
            methodParametr.add(new Integer(4));
            object = owner.menuAuto.A2;
        }
        if (e == 11) {
            y0 = 2;
            method = MenuAuto.AutomatMenuAuto.class.getMethod("A", classes);
            methodParametr.add(new Integer(11));
            object = owner.menuAuto.A2;
        }
        if (e == 111) {
            y0 = 1;
        }

        if (e == 12) {
            y0 = 3;
        }

        if (e == 13) {
            y0 = 4;
        }

        if (e == 7) {
            y0 = 2;
            method = BackGr.AutomatBackGr.class.getMethod("A", classes);
            methodParametr.add(new Integer(7));
```

```
                        object = owner.backGr.A1;
                    }
                    if (e == 8) {
                        y0 = 2;
                        method = BackGr.AutomatBackGr.class.getMethod("A", classes);
                        methodParametr.add(new Integer(8));
                        object = owner.backGr.A1;
                    }
                    break;

                case 3:
                    if (e == 1) {
                        y0 = 3;
                        method =
MenuChooseElement.AutomatMenuChooseElement.class.getMethod("A", classes);
                        methodParametr.add(new Integer(1));
                        object = owner.menuChooseSize.A3;
                    }
                    if (e == 2) {
                        y0 = 3;
                        method =
MenuChooseElement.AutomatMenuChooseElement.class.getMethod("A", classes);
                        methodParametr.add(new Integer(2));
                        object = owner.menuChooseSize.A3;
                    }
                    if (e == 3) {
                        y0 = 3;
                        method =
MenuChooseElement.AutomatMenuChooseElement.class.getMethod("A", classes);
                        methodParametr.add(new Integer(3));
                        object = owner.menuChooseSize.A3;
                    }
                    if (e == 4) {
                        y0 = 3;
                        method =
MenuChooseElement.AutomatMenuChooseElement.class.getMethod("A", classes);
                        methodParametr.add(new Integer(4));
                        object = owner.menuChooseSize.A3;
                    }

                    if (e == 11) {
                        y0 = 3;
                        method =
MenuChooseElement.AutomatMenuChooseElement.class.getMethod("A", classes);
                        methodParametr.add(new Integer(11));
                        object = owner.menuChooseSize.A3;
                    }
                    if (e == 7) {
                        y0 = 3;
                        method = BackGr.AutomatBackGr.class.getMethod("A", classes);
                        methodParametr.add(new Integer(7));
                        object = owner.backGr.A1;
                    }
                    if (e == 8) {
                        y0 = 3;
                        method = BackGr.AutomatBackGr.class.getMethod("A", classes);
                        methodParametr.add(new Integer(8));
                        object = owner.backGr.A1;
                    }
                    if (e == 111) {
                        y0 = 1;
                    }
                    break;

                case 4:
                    if (e == 1) {
```

```
                            y0 = 4;
                            method =
MenuChooseFlashMode.AutomatMenuChooseElement.class.getMethod("A", classes);
                            methodParametr.add(new Integer(1));
                            object = owner.menuChooseFlashMode.A3;
                        }
                        if (e == 2) {
                            y0 = 4;
                            method =
MenuChooseFlashMode.AutomatMenuChooseElement.class.getMethod("A", classes);
                            methodParametr.add(new Integer(2));
                            object = owner.menuChooseFlashMode.A3;
                        }
                        if (e == 3) {
                            y0 = 4;
                            method =
MenuChooseFlashMode.AutomatMenuChooseElement.class.getMethod("A", classes);
                            methodParametr.add(new Integer(3));
                            object = owner.menuChooseFlashMode.A3;
                        }
                        if (e == 4) {
                            y0 = 4;
                            method =
MenuChooseFlashMode.AutomatMenuChooseElement.class.getMethod("A", classes);
                            methodParametr.add(new Integer(4));
                            object = owner.menuChooseFlashMode.A3;
                        }

                        if (e == 11) {
                            y0 = 4;
                            method =
MenuChooseFlashMode.AutomatMenuChooseElement.class.getMethod("A", classes);
                            methodParametr.add(new Integer(11));
                            object = owner.menuChooseFlashMode.A3;
                        }
                        if (e == 7) {
                            y0 = 4;
                            method = BackGr.AutomatBackGr.class.getMethod("A", classes);
                            methodParametr.add(new Integer(7));
                            object = owner.backGr.A1;
                        }
                        if (e == 8) {
                            y0 = 4;
                            method = BackGr.AutomatBackGr.class.getMethod("A", classes);
                            methodParametr.add(new Integer(8));
                            object = owner.backGr.A1;
                        }
                        if (e == 111) {
                            y0 = 1;
                        }
                        break;

                    case 5:
                        if ((e == 15) && x1()) {
                            y0 = 1;
                        }
                        if ((e == 15) && x2()) {
                            y0 = 0;
                        }
                        break;

                    case 6:
                        if (e == 17) {
                            y0 = 1;
                            z0_61();
                        }
```

```
                default:
                    break;
            }

        } catch (java.lang.NoSuchMethodException exp) {
        }

        logTrans(y0, yold, e);

        setState(y0);

        if (y0 != yold) {
            switch (y0) {
                case 0:
                    z0_6();
                    break;
                case 1:
                    z0_1();
                    break;
                case 2:
                    z0_2();
                    break;
                case 3:
                    z0_3();
                    break;
                case 4:
                    z0_4();
                    break;
                case 5:
                    z0_5();
                    break;
                case 6:
                    z0_7();
                    break;
            }
        }

        if (method != null) {
            java.lang.Object parametrs[] = methodParametr.toArray();

            try {
                method.invoke(object, parametrs);
            } catch (java.lang.IllegalAccessException exp) {
            } catch (java.lang.reflect.InvocationTargetException exp) {}
        }

        owner.clientArea.repaint();

    };

};

public java.util.Timer timer = new java.util.Timer();
public class ShootTimerTask extends java.util.TimerTask {
    public void run() {
        A0.A(17);
    };
};
private ShootTimerTask shootTimerTask;

/**
 * Автомат, управляющий экраном
 */
private AutomatScreen A0;

private int x; //координаты экрана на панели
```

```
private int y;

private int width; //ширина и высота экрана
private int height;

public int getX() {
    return x;
};
public int getY() {
    return y;
};

public int getWidth() {
    return width;
};
public int getHeight() {
    return height;
};

//то, что установлено на барабане
private int mode; //0 - автосъемка, 1 - просмотр кадров; 2 - процесс фотографирования
private boolean onoff; //true -включен, false - выключен


public PhotoCamera owner; //приложение-владелец
private BackGr backGr; //автомат "фон"

/**
 * Конструктор с парамером o - приложение-владелец
 */
public Screen(PhotoCamera o) {
    owner = o;
    //cannonName = o.getName() + " cannon";
    A0 = new AutomatScreen();
    onoff = false;
    mode = 0;
    x = 10;
    y = 10;
    width = 200;
    height = 150;
};

/**
 * Метод для перерисовки экрана
 */
public void draw(Graphics g) {
    BackGr backGr = owner.backGr; //сначала перерисовываем фон,

    /*Перерисовка для случая процессасъемки*/
    if (A0.getState() == 6) {
        g.drawString("Busy...", 10, 10);
        return;
    }

    if (onoff) {
        backGr.draw(g);
        owner.menuAuto.draw(g);
        owner.menuChooseSize.draw(g);
        owner.menuChooseFlashMode.draw(g);
    }   //если, конечно, фотоаппарат включен

    g.drawRect(x, y, width, height); //дальше просто рисуем рамку
    Color oldc = g.getColor();
    g.setColor(Color.lightGray);
    for (int i = 1; i <= x + 1; i++) {
        g.drawRect(x - i, y - i, width + 2 * i, height + 2 * i);
```

```
    }
    ;
    g.setColor(oldc);
};

/**
 * Методы,посылающие сообщения автомату, управляющему экраном
 */
public void pressUp() {
    A0.A(1);
};

public void pressDown() {
    A0.A(2);
};

public void pressRight() {
    A0.A(3);
};

public void pressLeft() {
    A0.A(4);
};

public void pressZoomIn() {
    A0.A(7);
};

public void pressZoomOut() {
    A0.A(8);
};

public void pressMenubut() {
    A0.A(11); /*В документации, это событие e11*/
};

public void pressSlideView() {
    A0.A(10);
};

public void pressAutoMode() {
    A0.A(9);
};

public void pressOn() {
    A0.A(15);
};

public void pressOff() {
    A0.A(14);
};

public void pressEscapeMenubut() {
    A0.A(111); /*В документации, это событие e11_1*/
};

public void pressSetupMenubut() {
    A0.A(110); /*В документации, это событие e11_0*/
};

public void pressMenuSizeChoosen() {
    A0.A(12);
};

public void pressMenuFlashChoosen() {
    A0.A(13);
```

```
    };

    public void shoot() {
        A0.A(16);

    };
}

;
```

BackGr. java

```
import java.awt.*;
import java.util.*;
import java.awt.geom.*;
import java.lang.reflect.Method;
import java.io.*;

/**
* класс "Фон"
*/
public class BackGr{

    /**
    * Автомат, управляющий фоном.
    */
    class AutomatBackGr extends Automat{

        public AutomatBackGr() {
            setState(3);
            setName("Automat A1");
        };


        /**
        * Функции-выходы
        */
        public void z1_01() {
            slideNum++;
            if (slideNum == slideTotal + 1) slideNum = 1;
        };

        public void z1_02() {
            slideNum--;
            if (slideNum == 0) slideNum = slideTotal;
        };

        public void z1_11() {
            slideNum++;
            if (slideNum == slideTotal + 1) slideNum = 1;
        };

        public void z1_12() {
            slideNum--;
            if (slideNum == 0) slideNum = slideTotal;
        };

        public void z1_2() {
            zoom = 2;
            shiftX = (int)(imgWidth/2);
            shiftY = (int)(imgHeight/2);
        };

        public void z1_1() {
```

```
            zoom = 1;
            shiftX = 0;
            shiftY = 0;
        };

        public void z1_0() {
            //
        };

        public void z1_21() {
            int oldzoom = zoom;
            if (zoom < 4) zoom++;
            shiftX = (int)((shiftX + (double)(width)/2)*((double)(zoom)/(double)(oldzoom)) -
(double)(width)/2);
            shiftY = (int)((shiftY + (double)(height)/2)*((double)(zoom)/(double)(oldzoom)) -
(double)(height)/2);
            if (shiftX < 0) shiftX = 0;
            if (shiftY < 0) shiftY = 0;
        };

        public void z1_22() {
            shiftX = shiftX - 50;
            if (shiftX < 0) shiftX = 0;
        };

        public void z1_23() {
            shiftX = shiftX + 50;
            if (shiftX > imgWidth * zoom - imgWidth) shiftX = imgWidth * zoom - imgWidth;
        };

        public void z1_25() {
            shiftY = shiftY - 50;
            if (shiftY < 0) shiftY = 0;
        };

        public void z1_24() {
            shiftY = shiftY + 50;
            if (shiftY > imgHeight * zoom - imgHeight) shiftY = imgHeight * zoom - imgHeight;
        };

        public void z1_3() {
            //zoom = 1;
            shiftX = (int)((zoom - 1) * (double)(imgWidth)/2);
            shiftY = (int)((zoom - 1) * (double)(imgHeight)/2);
        };

        public void z1_31() {
            if (zoom < 4) zoom++;
            shiftX = (int)((zoom - 1) * (double)(imgWidth)/2);
            shiftY = (int)((zoom - 1) * (double)(imgHeight)/2);
        };

        public void z1_32() {
            if (zoom > 1) zoom--;
            shiftX = (int)((zoom - 1) * (double)(imgWidth)/2);
            shiftY = (int)((zoom - 1) * (double)(imgHeight)/2);
        };

        public void z1_33() {
            //
            owner.pressSetupMenubut();
        };

        /**
        * Функция автомата
        */
```

```
public void A(Integer event) {
    A(event.intValue());
};

public void A(int e) {
    int y0 = getState();
    int yold = y0;

    Method method = null;
    Vector methodParametr = new Vector();
    Object object = null;
    Class classes[] = new Class[0];

    try {
        switch (y0) {
            case 0:
                if (e == 3) {
                    y0 = 0;
                    z1_01();
                }
                if (e == 4) {
                    y0 = 0;
                    z1_02();
                }
                if (e == 7) {
                    y0 = 1;
                }
                if (e == 9) {
                    y0 = 3;
                }
                if (e == 10) {
                    y0 = 1;
                    z1_1();
                }
                break;

            case 1:
                if (e == 3) {
                    y0 = 1;
                    z1_11();
                }
                if (e == 4) {
                    y0 = 1;
                    z1_12();
                }
                if (e == 7) {
                    y0 = 2;
                    z1_2();
                }
                if (e == 8) {
                    y0 = 0;
                    z1_0();
                }
                if (e == 9) {
                    y0 = 3;
                }
                if (e == 10) {
                    y0 = 1;
                    z1_1();
                }
                break;

            case 2:
                if (e == 8) {
                    y0 = 1;
                    z1_1();
```

```
            }
            if (e == 1) {
                y0 = 2;
                z1_25();
            }
            if (e == 2) {
                y0 = 2;
                z1_24();
            }
            if (e == 3) {
                y0 = 2;
                z1_23();
            }
            if (e == 4) {
                y0 = 2;
                z1_22();
            }
            if (e == 7) {
                y0 = 2;
                z1_21();
            }
            if (e == 9) {
                y0 = 3;
            }
            if (e == 10) {
                y0 = 1;
                z1_1();
            }
            break;

        case 3:
            if (e == 7) {
                y0 = 3;
                z1_31();
            }
            if (e == 8) {
                y0 = 3;
                z1_32();
            }
            if (e == 11) {
                y0 = 3;
                method = BackGr.AutomatBackGr.class.getMethod("z1_33", classes);
                object = this;
            }
            if (e == 9) {
                y0 = 3;
            }
            if (e == 10) {
                y0 = 1;
                z1_1();
            }
            break;

        default:
            break;
    }
} catch (java.lang.NoSuchMethodException exp) {
}

logTrans(y0, yold, e);
setState(y0);


if (y0 != yold)
switch (y0) {
    case 3:
```

```
                z1_3();
                break;
        }

        if (method != null) {
            java.lang.Object parametrs[] = methodParametr.toArray();

            try {
                method.invoke(object, parametrs);
            } catch (java.lang.IllegalAccessException exp) {
            } catch (java.lang.reflect.InvocationTargetException exp) {}
        }
    };


};

/**
* Автомат, управляющий фоном.
*/
public AutomatBackGr A1;

/**
* Экран-владелец
*/
private Screen owner;

/**
* Координаты верхнего левого угла фона на панели
*/
private int x;
private int y;

/**
* Ширина и высота
*/
private int width;
private int height;

/**
* Стандартная ширина и высота изображения (200x150)
*/
private int imgWidth;
private int imgHeight;

/**
* Просто изображение (кучу раз используется в перерисовке)
*/
private Image img;

/*Физические параметры*/
/**
* Номер просматриваемого снимка
*/
private int slideNum;

/**
* Всего количество снимков
*/
private int slideTotal;
public int getSlideTotal() {return slideTotal;};
public void setSlideTotal(int n) {slideTotal = n;};
```

```
/**
* Текущая величина зума
*/
private int zoom;

public int getZoom() {return zoom;};

/**
* Скроллинг по горизонтали
*/
private int shiftX;

/**
* Скроллинг по вертикали
*/
private int shiftY;

/**
* Конструктор
*/
public BackGr(Screen s) {
    //owner = o;
    //cannonName = o.getName() + " cannon";
    A1 = new AutomatBackGr();
    owner = s;
    x = s.getX() + 1;
    y = s.getY() + 1;
    width = s.getWidth() - 2;
    height = s.getHeight() - 2;
  img = ImageMap.getImage("img/img.gif");

  imgWidth = 200;
  imgHeight = 150;

  slideNum = 1;

  /*slideTotal = 7;*/
  File file = new File(System.getProperty("user.dir") + "/img");

  FilenameFilter filter = new FilenameFilter() {
      public boolean accept(File dir, String name) {
          File file = new File(dir.toString() + "/" + name);
          return (file.exists() && file.isFile());
      };
  };

  slideTotal = file.list(filter).length;

  zoom = 1;
  shiftX = 0;
  shiftY = 0;
};

/**
* Метод для перерисовки фона
*/
public void draw(Graphics g) {
    int state = A1.getState();
    int xx;
    int yy;
    int halfWidth = (int)((double)(imgWidth)/2);
    int halfHeight = (int) ((double) (imgHeight) / 2);

    //если состояние 1 или 2, то перерисовываем один кадр, возможно с зумом
    if ((state == 1) || (state == 2)) {
```

```
    img = ImageMap.getImage("img/" + slideNum + "img.gif");

    Graphics2D g2 = (Graphics2D) g;

    AffineTransform aT = g2.getTransform();

    double sx;
    double sy;
    switch (zoom) {
        case 1:
            sx = 0.25;
            sy = 0.25;
            break;

        case 2:
            sx = 0.5;
            sy = 0.5;
            break;

        case 3:
            sx = 0.75;
            sy = 0.75;
            break;

        case 4:
            sx = 1;
            sy = 1;
            break;

        default:
            sx = 0;
            sy = 0;
            break;
    }

    AffineTransform scaleTransform = AffineTransform.getScaleInstance(sx, sy);

    g2.transform(scaleTransform);
    g2.drawImage(img, (int) ((x - shiftX) / sx), (int) ((y - shiftY) / sy), null);

    g2.setTransform(aT);

    //Если зум = 1 то еще добавим надпись
    if (zoom == 1) {
        g.drawString("Просмотр: pic000" + slideNum + ".jpg", x + 10, y + 20);
    }

}

//в состоянии 0 перерисовываем целую галлерею кадров
if (state == 0) {
    int num;
    int colnum = 2;
    int rownum = (int) ((double) (slideTotal + 1) / (colnum));
    int increase = (int) ((slideNum - 1) / 4) * 4;
    for (int i = 1; i <= rownum; i++) {
        for (int j = 1; j <= colnum; j++) {
            num = (i - 1) * colnum + j + increase; //!
            if (num <= slideTotal) {
                Color old = g.getColor();

                Graphics2D g2 = (Graphics2D) g;
                AffineTransform aT = g2.getTransform();
                double sx = 0.125;
                double sy = 0.125;
                AffineTransform scaleTransform = AffineTransform.getScaleInstance(sx,
```

```
                                                                                sy);
                        g2.transform(scaleTransform);

                        img = ImageMap.getImage("img/" + num + "img.gif");

                        xx = (j - 1) * halfWidth;
                        yy = (i - 1) * halfHeight;
                        g2.drawImage(img, (int)((x + xx) / sx), (int)((y + yy) / sx), null);

                        g2.setTransform(aT);
                        if (num == slideNum) {

                            g.setColor(Color.blue);
                            g.drawRect(x + xx, y + yy, halfWidth - 1, halfHeight - 1);
                            g.drawRect(x + xx + 1, y + yy + 1, halfWidth - 3, halfHeight - 3);


                        }
                        ;
                        g.setFont(Font.getFont("Courier"));
                        g.setColor(Color.black);
                        g.drawString("" + num, x + xx + 10, y + yy + 20);
                        g.setColor(old);
                    }
                    ;
                }
                ;
            }
            ;
        }

        // в состоянии 3 рисуем видоискатель
        if (state == 3) {
            img = ImageMap.getImage("img/sight/sight" + zoom + ".gif");
            g.drawImage(img, x - shiftX, y - shiftY, null);
            g.drawString("Видоискатель", x + 10, y + 20);
            g.drawString("Вспышка: " + owner.owner.menuChooseFlashMode.getSelectedItem(), x +
10, y + 40);
            g.drawString(owner.owner.menuChooseSize.getSelectedItem(), x + 10, y + 60);
        }
    };

};
```

MenuAuto. java

```
import java.awt.*;
import java.util.*;
import java.lang.reflect.Method;

/**
 * класс "Меню настройки режима авто"
 */
public class MenuAuto {

    /**
     * Класс "Автомат для управления меню"
     */
    class AutomatMenuAuto extends Automat {

        /**
         * Конструктор
         */
        public AutomatMenuAuto() {
```

```
    setState(0);
    setName("Automat A2");
};


/**
 * Функция-флажок. Просто говорит, верно ли, что меню отображается на экране
 */
public boolean x1() {
    return (onoff == true);
};

/**
 * Функции-выходы
 */
public void z2_0() {
    owner.pressEscapeMenubut();
    onoff = false;
};

public void z2_01() {
    choosenBut = 0;
    onoff = true;
};


public void z2_11() {
    //owner.pressMenuFlashChoosen();
    onoff = true;
    choosenBut = 2;

};
public void z2_12() {
    //owner.pressMenuSizeChoosen();
    onoff = true;
    choosenBut = 1;
};

public void z2_10() {
    onoff = false;
    if (choosenBut == 2) {
        owner.pressMenuFlashChoosen();
    }else{
        owner.pressMenuSizeChoosen();
    }
};

public void A(Integer event) {
    A(event.intValue());
}

/**
 * Непосредственно функция автомата
 */
public void A(int e) {
    int y0 = getState();
    int yold = y0;

    Method method = null;
    Vector methodParametr = new Vector();
    Object object = null;
    Class classes[] = new Class[0];

    try {
        switch (y0) {
            case 0:
```

```
                        if ((e == 11) && (!x1())) {
                            y0 = 0;
                            method = MenuAuto.AutomatMenuAuto.class.getMethod("z2_01",
classes);
                            object = this;
                        } else if ((e == 11) && (x1())) {
                            y0 = 0;
                            method = MenuAuto.AutomatMenuAuto.class.getMethod("z2_0",
classes);
                            object = this;
                        } else if (e == 3) {
                            y0 = 1;
                            method = MenuAuto.AutomatMenuAuto.class.getMethod("z2_11",
classes);
                            object = this;
                        } else if (e == 4) {
                            y0 = 1;
                            method = MenuAuto.AutomatMenuAuto.class.getMethod("z2_12",
                                                                              classes);
                            object = this;
                        }
                        break;

                    case 1:
                        if ((e == 11) && (!x1())) {
                            y0 = 0;
                            method = MenuAuto.AutomatMenuAuto.class.getMethod("z2_01",
                                                                              classes);
                            object = this;
                        } else if ((e == 11) && (x1())) {
                            y0 = 0;
                            method = MenuAuto.AutomatMenuAuto.class.getMethod("z2_10",
                                                                              classes);
                            object = this;
                        } else if (e == 3) {
                            y0 = 1;
                            method = MenuAuto.AutomatMenuAuto.class.getMethod("z2_11",
                                                                              classes);
                            object = this;
                        } else if (e == 4) {
                            y0 = 1;
                            method = MenuAuto.AutomatMenuAuto.class.getMethod("z2_12",
                                                                              classes);
                            object = this;
                        }
                        break;

                    default:
                        break;
                }

            } catch (java.lang.NoSuchMethodException exp) {
            }
            setState(y0);
            logTrans(y0, yold, e);


            if (method != null) {
                java.lang.Object parametrs[] = methodParametr.toArray();

                try {
                    method.invoke(object, parametrs);
                } catch (java.lang.IllegalAccessException exp) {
                } catch (java.lang.reflect.InvocationTargetException exp) {}
            }
        };
```

```
    };

    /**
     * Автомат, управляющий меню
     */
    public AutomatMenuAuto A2;

    private boolean onoff; //true -включен, false - выключен
    private int choosenBut; //номер выбранной кнопки

    /**
     * Экран-владелец
     */
    private Screen owner;
//    private MyApplet owner; //приложение-владелец
//    private BackGr backGr; //автомат "фон"

    /**
     * Координаты верхнего левого угла фона на панели
     */
    private int x;
    private int y;

    /**
     * Ширина и высота
     */
    private int width;
    private int height;

    /**
     * Конструктор с парамером o - приложение-владелец
     */
    public MenuAuto(Screen o) {
        owner = o;
        A2 = new AutomatMenuAuto();

        x = o.getX() + 1;
        y = o.getY() + 1;
        width = o.getWidth() - 2;
        height = o.getHeight() - 2;

        onoff = false;
        choosenBut = 0;
    };

    private void drawMenuItem(int n, boolean ifSelected, Graphics g) {
        int width = 70;
        int height = 30;
        String caption;
        if (n == 1) {
            caption = "ISO";
        } else {
            caption = "Flash";
        }
        ;
        int x = 30 + (n - 1) * (width + 20);
        int y = 60;

        Color oldc = g.getColor();
        g.setColor(Color.darkGray);
        g.fillRect(x, y, width, height);

        g.setColor(Color.white);
        if (ifSelected) g.setColor(Color.yellow);
```

```
        g.drawRect(x, y, width, height);

        int textWidth = g.getFontMetrics().stringWidth(caption);
        g.drawString(caption, x + (width - textWidth)/2, y + (int) (height / 2) + 5);

        g.setColor(oldc);
    };

    /**
     * Метод для перерисовки экрана
     */
    public void draw(Graphics g) {
        if (onoff) {
            Color oldc = g.getColor();
            g.setColor(Color.black);

            drawMenuItem(1, (choosenBut == 1), g);
            drawMenuItem(2, (choosenBut == 2), g);
            g.setColor(oldc);
        }
    };

    /**
     * Методы,посылающие сообщения автомату, управляющему экраном
     */
    public void pressMenubut() {
        A2.A(11);
    };

    public void pressRight() {
        A2.A(3);
    };

    public void pressLeft() {
        A2.A(4);
    };


}
```

MenuChooseSize. java

```
public class MenuChooseSize extends MenuChooseElement{
    public MenuChooseSize(Screen screen) {
        super(screen);
        setItem(0, "ISO100");
        setItem(1, "ISO200");
        setItem(2, "ISO400");
    }
}
```

MenuChooseFlashMode. java

```
public class MenuChooseFlashMode extends MenuChooseElement{
    public MenuChooseFlashMode(Screen screen) {
        super(screen);
        setItem(0, "Auto");
        setItem(1, "On");
        setItem(2, "Off");
```

```
    }
}
```

Automat. java

```
/**
 * <code>Automat</code>
 * Базовый класс для любого автомата. Содержит объявление необходимых методов.
 */
public abstract class Automat{

    /**
    * Номер состояния автомата
    */
    private int y0;

    /**
    * Возвращает номер состояния автомата
    */
    protected int getState() {return y0;};

    /**
    * Устанавливает новое состояния автомата
    */
    protected void setState(int y) {y0 = y;};

    /**
    * Имя автомата
    */
    private String name;

    /**
    * Получить имя автомата
    */
    protected String getName() {return name;};

    /**
    * Установить имя автомата
    */
    protected void setName(String n) {name = n;};

    /**
    * Тип автомата (A0, A1, A2 и.т.д...)
    */
    private String automatType;

    /**
    * Получить тип автомата
    */
    protected String getType() {return automatType;};

    /**
    * Установить тип автомата
    */
    protected void setType(String t) {automatType = t;};

    /**
    * Выводит лог о том, что автомат перешел
    * в состояние to из состояния from, получив событие e
    */
    protected void logTrans(int to, int from, int e) {
        java.util.Date time = new java.util.Date();
        time = java.util.Calendar.getInstance(java.util.Locale.getDefault()).getTime();
        Log.writeln(time.toString() + "| " + "T " + automatType +
```

```
            " <" + name + ">: has changed state from " + from + " to " + to + " | event: " + e);
    };

    /**
    * Функция автомата. Должна осуществлять переход автомата, получившего событие e
    * в другое состояние
    */
    public abstract void A(int e);
};
```

ClientArea. java

```
import java.awt.*;

class ClientArea extends Panel {
         public Image buffer;
         private Image image;
         private Graphics bg;

         public PhotoCamera owner;

         public ClientArea(PhotoCamera o) {
             super();
             owner = o;
         }
         public void paint(Graphics g) {
                int clientWidth  = getSize().width;
                int clientHeight = getSize().height;

                if ((buffer == null)/*||(owner.flagChangeSize)*/) {
                    buffer = this.createImage(clientWidth, clientHeight);
                }else
                {
                    buffer.flush();
                }

                if ((image == null)/*||(owner.flagChangeSize)*/) {
                    image = this.createImage(clientWidth, clientHeight);
                }

                bg = buffer.getGraphics();

                Graphics bbg = image.getGraphics();

                bbg.drawImage(buffer, 0, 0, null);

                owner.paintClient(bbg);

                g.drawImage(image, 0, 0, null);
                bg.dispose();
         }
         public final void update(Graphics g) {
             paint(g);
         }

    };
```

ControlPanel. java

```
import java.awt.*;
import java.awt.event.*;
```

```
import javax.swing.*;

import java.util.*;

/**
* Класс для работы с панелью управления
*/
public class ControlPanel extends JPanel implements ActionListener{

    public JButton up;
    public JButton down;
    public JButton left;
    public JButton right;

    public JButton menubut;

    public ControlPanel() {
        super();
        int butSize = 40;
        setBounds(245, 80, 10 * 4 + 3 * butSize, 10 * 4 + 3 * butSize);

        up = new JButton(ImageMap.getImageIcon("buttons/up.jpg"));
        up.setBounds(10 + butSize + 10, 10, butSize, butSize);

        down = new JButton(ImageMap.getImageIcon("buttons/down.jpg"));
        down.setBounds(10 + butSize + 10, 10 + 2*(butSize + 10), butSize, butSize);

        left = new JButton(ImageMap.getImageIcon("buttons/left.jpg"));
        left.setBounds(10,  10 + butSize + 10, butSize, butSize);
        right = new JButton(ImageMap.getImageIcon("buttons/right.jpg"));
        right.setBounds(10 + 2*(butSize + 10), 10 + butSize + 10, butSize, butSize);

        menubut = new JButton(ImageMap.getImageIcon("buttons/ok.jpg"));
        menubut.setBounds(10 + butSize + 10, 10 + butSize + 10, butSize, butSize);

        addActionListener(this);

        setLayout(null);

        add(up);
        add(down);
        add(right);
        add(left);
        add(menubut);
    };

    public void addActionListener(ActionListener listener) {
        up.addActionListener(listener);
        down.addActionListener(listener);
        right.addActionListener(listener);
        left.addActionListener(listener);
        menubut.addActionListener(listener);
    }

    public void addKeyListener(KeyListener listener) {
        super.addKeyListener(listener);
    }

    public void actionPerformed(ActionEvent e)     {
        Object source = e.getSource();

    };

}
```

TopPanel. java

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

import java.util.*;

/**
* Класс для работы с панелью управления
*/
public class TopPanel extends JPanel implements ActionListener{

    public JButton zoomIn;
    public JButton zoomOut;

    public JButton shoot;

    public JButton powerOn;
    public JButton powerOff;

    public JButton viewMode;
    public JButton autoMode;

    public TopPanel() {
        super();

        int onoffButSize = 40; //Размер кнопок "включить" и "выключить"
        int modeButSize = 40; //Размер кнопок "режим просмотра" и "режим авто"
        int zoomButSize = 40; //Размер кнопок "zoomIn-Out"

        int viewModeButX = 10 + onoffButSize + 40;

        int autoModeButX = viewModeButX + 30 + modeButSize;

        viewMode = new JButton(ImageMap.getImageIcon("buttons/view.jpg"));
        viewMode.setBounds(autoModeButX, 20, modeButSize, modeButSize);


        autoMode = new JButton(ImageMap.getImageIcon("buttons/nature.jpg"));
        autoMode.setBounds(viewModeButX, 20, modeButSize, modeButSize);

        int zoomInButX = autoModeButX + modeButSize + 30;

        zoomOut = new JButton(ImageMap.getImageIcon("buttons/T.jpg"));
        zoomOut.setBounds(zoomInButX, 20, zoomButSize, zoomButSize);

        shoot = new JButton(ImageMap.getImageIcon("buttons/exclam.jpg"));
        shoot.setBounds(zoomInButX + zoomButSize + zoomButSize + 10 + 30, 20, zoomButSize,
zoomButSize);

        zoomIn = new JButton(ImageMap.getImageIcon("buttons/W.jpg"));
        zoomIn.setBounds(zoomInButX + zoomButSize + 10, 20, zoomButSize, zoomButSize);

        powerOn = new JButton(ImageMap.getImageIcon("buttons/on.jpg"));
        powerOn.setBounds(20, 19, onoffButSize, onoffButSize - 18);

        powerOff = new JButton(ImageMap.getImageIcon("buttons/off.jpg"));
        powerOff.setBounds(20, 19 + onoffButSize - 18, onoffButSize, onoffButSize - 18);

        addActionListener(this);

        setLayout(null);

        add(powerOn);
```

```
        add(powerOff);
        add(shoot);
        add(zoomIn);
        add(zoomOut);

        add(viewMode);
        add(autoMode);
    };

    public void addActionListener(ActionListener listener) {
        powerOn.addActionListener(listener);
        powerOff.addActionListener(listener);
        zoomIn.addActionListener(listener);
        zoomOut.addActionListener(listener);
        viewMode.addActionListener(listener);
        autoMode.addActionListener(listener);
        shoot.addActionListener(listener);
    }

    public void addKeyListener(KeyListener listener) {
        super.addKeyListener(listener);
        powerOn.addKeyListener(listener);
    }

    public void actionPerformed(ActionEvent e)     {
        Object source = e.getSource();

    };

}
```

ImageMap. java

```
import java.awt.*;
import javax.swing.*;
import java.util.*;


abstract public class ImageMap {
       protected static Map imageMap = new HashMap(); // хэш картинок
       protected static Map cursorMap = new HashMap(); // хэш курсоров

       protected static class CursorInfo {
              public String path;
              public Point center;
              public String name;
              public CursorInfo(String tpath, Point tcenter, String tname) {
                     path = tpath;
                     center = tcenter;
                     name = tname;
              }
       };

    /**
     * Берем ImageIcon с путем path
     */
    public static ImageIcon getImageIcon(String path) {
              if (imageMap.containsKey(path))
                     return (ImageIcon)imageMap.get(path);
              ImageIcon img = new ImageIcon(path);
              imageMap.put(path, img);
              return img;
    }
```

```
    /**
     * Берем Image с путем path
     */
    public static Image getImage(String path) {
      return getImageIcon(path).getImage();
    }


      public static Cursor getCursor(String path, Point center, String name) {
            CursorInfo ci = new CursorInfo(path, center, name);
            if (cursorMap.containsKey(ci))
                  return (Cursor)cursorMap.get(ci);
            Cursor cursor =  Toolkit.getDefaultToolkit().createCustomCursor(
                                                  getImage(path),
                                 center,
                                 name);
        cursorMap.put(ci, cursor);
        return cursor;
      }

}
```

MenuChooseElement. java

```
import java.lang.reflect.Method;
import java.util.Vector;
import java.awt.*;

public class MenuChooseElement {
    /**
     * Класс "Автомат для управления меню"
     */
    public class AutomatMenuChooseElement extends Automat {

        /**
         * Конструктор
         */
        public AutomatMenuChooseElement() {
            setState(0);
            setName("Automat A3");
        };


        /**
         * Функция-флажок. Просто говорит, верно ли, что меню отображается на экране
         */
        public boolean x1() {
            return (onoff == true);
        };

        /**
         * Функции-выходы
         */
        public void z3_0() {
            owner.pressEscapeMenubut();
            onoff = false;
        };

        public void z3_00() {
            //owner.pressEscapeMenubut();
            onoff = true;
        };

        public void z3_01() {
```

```
            onoff = false;
            owner.pressEscapeMenubut();
        };

        public void z3_11() {
            onoff = true;
            choosenBut = 1;
        };

        public void z3_21() {
            onoff = true;
            choosenBut = 2;
        };

        public void z3_31() {
            onoff = true;
            choosenBut = 3;
        };

        public void A(Integer event) {
            A(event.intValue());
        }
        /**
         * Непосредственно функция автомата
         */
        public void A(int e) {
            int y0 = getState();
            int yold = y0;

            Method method = null;
            Vector methodParametr = new Vector();
            Object object = null;
            Class classes[] = new Class[0];

            try {
                switch (y0) {
                    case 0:
                        if ((e == 2) || (e == 1)) {
                            y0 = 1;
                            method =
MenuChooseElement.AutomatMenuChooseElement.class.getMethod("z3_11", classes);
                            object = this;
                        } else if ((e == 11) && (x1())) {
                            y0 = 0;
                            method =
MenuChooseElement.AutomatMenuChooseElement.class.getMethod("z3_01", classes);
                            object = this;
                        } else if ((e == 11) && (!x1())) {
                            y0 = 0;
                            method =
MenuChooseElement.AutomatMenuChooseElement.class.getMethod("z3_00", classes);
                            object = this;
                        } else if (e == 4) {
                            y0 = 0;
                            method =
MenuChooseElement.AutomatMenuChooseElement.class.getMethod("z3_0", classes);
                            object = this;
                        }
                        break;

                    case 1:
                        if (e == 1) {
                            y0 = 3;
                            method =
MenuChooseElement.AutomatMenuChooseElement.class.getMethod("z3_31", classes);
                            object = this;
```

```
                        } else if (e == 2) {
                            y0 = 2;
                            method =
MenuChooseElement.AutomatMenuChooseElement.class.getMethod("z3_21", classes);
                            object = this;
                        } else if ((e == 11) && (x1())) {
                            y0 = 0;
                            method =
MenuChooseElement.AutomatMenuChooseElement.class.getMethod("z3_01", classes);
                            object = this;
                        } else if ((e == 11) && (!x1())) {
                            y0 = 0;
                            method =
MenuChooseElement.AutomatMenuChooseElement.class.getMethod("z3_00", classes);
                            object = this;
                        } else if (e == 4) {
                            y0 = 0;
                            method =
MenuChooseElement.AutomatMenuChooseElement.class.getMethod("z3_0", classes);
                            object = this;
                        }
                        break;

                    case 2:
                        if (e == 1) {
                            y0 = 1;
                            method =
MenuChooseElement.AutomatMenuChooseElement.class.getMethod("z3_11", classes);
                            object = this;
                        } else if (e == 2) {
                            y0 = 3;
                            method =
MenuChooseElement.AutomatMenuChooseElement.class.getMethod("z3_31", classes);
                            object = this;
                        } else if ((e == 11) && (x1())) {
                            y0 = 0;
                            method =
MenuChooseElement.AutomatMenuChooseElement.class.getMethod("z3_01", classes);
                            object = this;
                        } else if ((e == 11) && (!x1())) {
                            y0 = 0;
                            method =
MenuChooseElement.AutomatMenuChooseElement.class.getMethod("z3_00", classes);
                            object = this;
                        } else if (e == 4) {
                            y0 = 0;
                            method =
MenuChooseElement.AutomatMenuChooseElement.class.getMethod("z3_0", classes);
                            object = this;
                        }
                        break;

                    case 3:
                        if (e == 1) {
                            y0 = 2;
                            method =
MenuChooseElement.AutomatMenuChooseElement.class.getMethod("z3_21", classes);
                            object = this;
                        } else if (e == 2) {
                            y0 = 1;
                            method =
MenuChooseElement.AutomatMenuChooseElement.class.getMethod("z3_11", classes);
                            object = this;
                        } else if ((e == 11) && (x1())) {
                            y0 = 0;
```

```
                            method =
MenuChooseElement.AutomatMenuChooseElement.class.getMethod("z3_01", classes);
                            object = this;
                        } else if ((e == 11) && (!x1())) {
                            y0 = 0;
                            method =
MenuChooseElement.AutomatMenuChooseElement.class.getMethod("z3_00", classes);
                            object = this;
                        } else if (e == 4) {
                            y0 = 0;
                            method =
MenuChooseElement.AutomatMenuChooseElement.class.getMethod("z3_0", classes);
                            object = this;
                        }
                        break;


                    default:
                        break;
                }

            } catch (java.lang.NoSuchMethodException exp) {
            }
            setState(y0);
            logTrans(y0, yold, e);

            if (method != null) {
                java.lang.Object parametrs[] = methodParametr.toArray();

                try {
                    method.invoke(object, parametrs);
                } catch (java.lang.IllegalAccessException exp) {
                } catch (java.lang.reflect.InvocationTargetException exp) {}
            }
        };
    };

    private Screen owner;

    private boolean onoff; //true -включен, false - выключен
    private int choosenBut; //номер выбранной кнопки

    private int itemsNum;
    public int getItemsNum() {return itemsNum;};

    private String[] items;
    public void setItem(int index, String value) {
        items[index] = value;
    }
    public String getItem(int index) {return items[index]; };
    public int getSelectedItemIndex() {
         return choosenBut - 1;
    };
    public String getSelectedItem() {
        if (choosenBut > 0) {
            return items[choosenBut - 1];
        };
        return "";
    };

    /**
     * Автомат, управляющий меню
     */
    public MenuChooseElement.AutomatMenuChooseElement A3;

    public MenuChooseElement(Screen screen) {
```

```
        owner = screen;
        A3 = new MenuChooseElement.AutomatMenuChooseElement();


        onoff = false;
        choosenBut = 1;
        itemsNum = 3;
        items = new String[itemsNum];
    };

    private void drawMenuItem(int n, boolean ifSelected, Graphics g) {
        int width = 70;
        int height = 20;
        int sizeBetweenItems = 10;
        String caption = items[n];

        int x = (owner.getWidth() - width) / 2 + owner.getX();
        int y = (owner.getHeight() - (sizeBetweenItems * (itemsNum - 1) + height * itemsNum))
/ 2 + owner.getY() +
                (n) * (height + sizeBetweenItems);

        Color oldColor = g.getColor();
        g.setColor(Color.darkGray);
        g.fillRect(x, y, width, height);

        g.setColor(Color.white);
        if (ifSelected) g.setColor(Color.yellow);
        g.drawRect(x, y, width, height);

        int textWidth = g.getFontMetrics().stringWidth(caption);
        g.drawString(caption, x + (width - textWidth)/2, y + (int) (height / 2) + 5);

        g.setColor(oldColor);
    };

    /**
     * Метод для перерисовки экрана
     */
    public void draw(Graphics g) {
        if (onoff) {
            Color oldc = g.getColor();
            g.setColor(Color.black);

            for (int i = 0; i <= itemsNum - 1; i++) {
                drawMenuItem(i, (choosenBut == i + 1)&&(A3.getState() != 0), g);
            }
            g.setColor(oldc);
        }
    };

    public void pressUp() {
        A3.A(1);
    };

    public void pressDown() {
        A3.A(2);
    };

    public void pressRight() {
        A3.A(3);
    };

    public void pressLeft() {
        A3.A(4);
    };
```

```
    public void pressMenubut() {
        A3.A(11); /*В документации, это событие e11*/
    };
}
```

Log. java

```
import java.io.*;
/**
 * Класс поддержки ведения логфайла
 */
public class Log {
     public static FileOutputStream fileOutputStream;

     public synchronized static void write(String l) {
         try{
             for (int i = 0; i < l.length(); i++) {
                 fileOutputStream.write((byte)(l.charAt(i)));
             };
         }catch (IOException e) { };
     };

     public synchronized static void writeln(String l) {
         try{
             for (int i = 0; i < l.length(); i++) {
                 fileOutputStream.write((byte)(l.charAt(i)));
             };
             fileOutputStream.write(13);
         }catch (IOException e) { };
     };

     static {
         try{
            fileOutputStream = new FileOutputStream(new File("log.txt"));
         }catch (FileNotFoundException e) { };
     }
}
```