

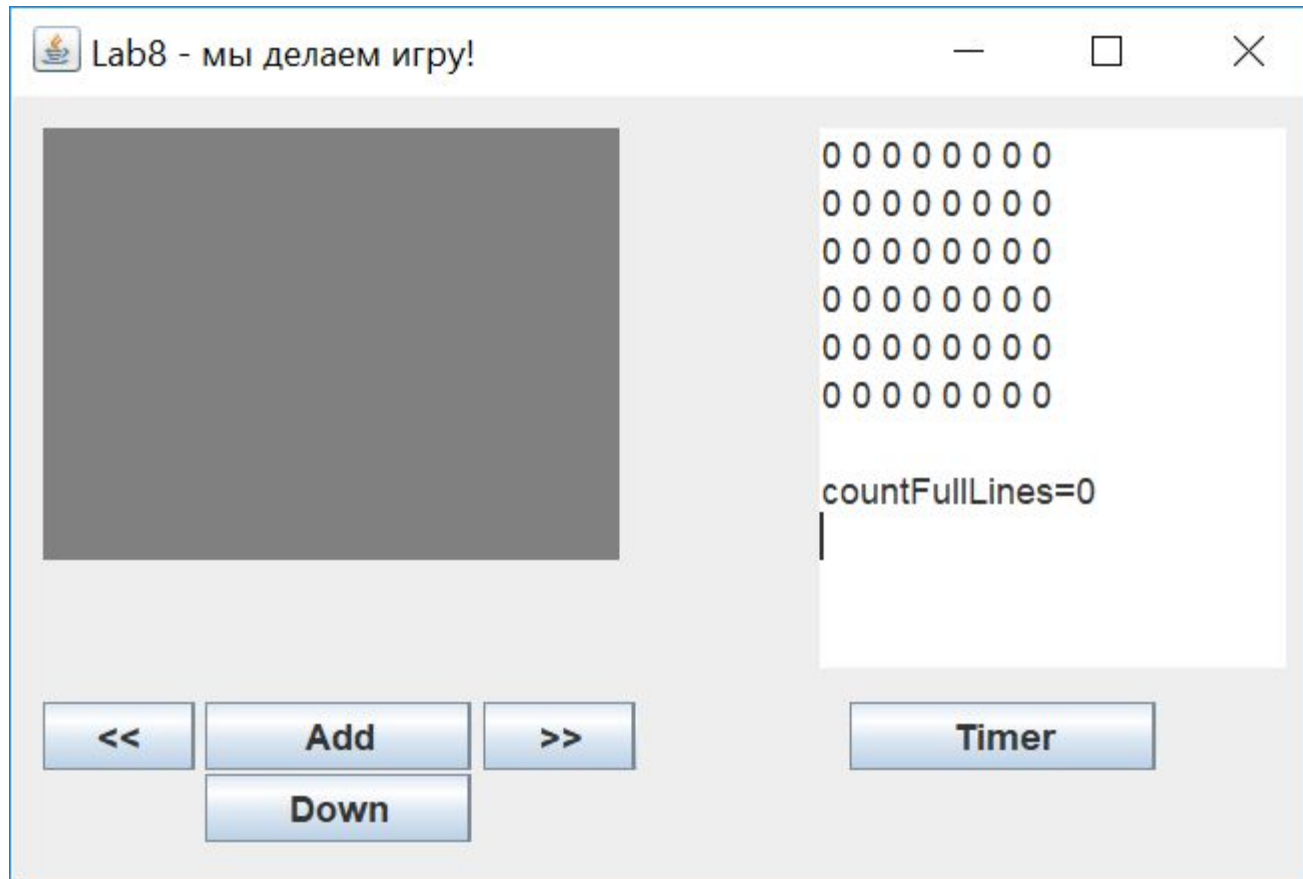
Основы программирования - Java

ФИСТ 1 курс

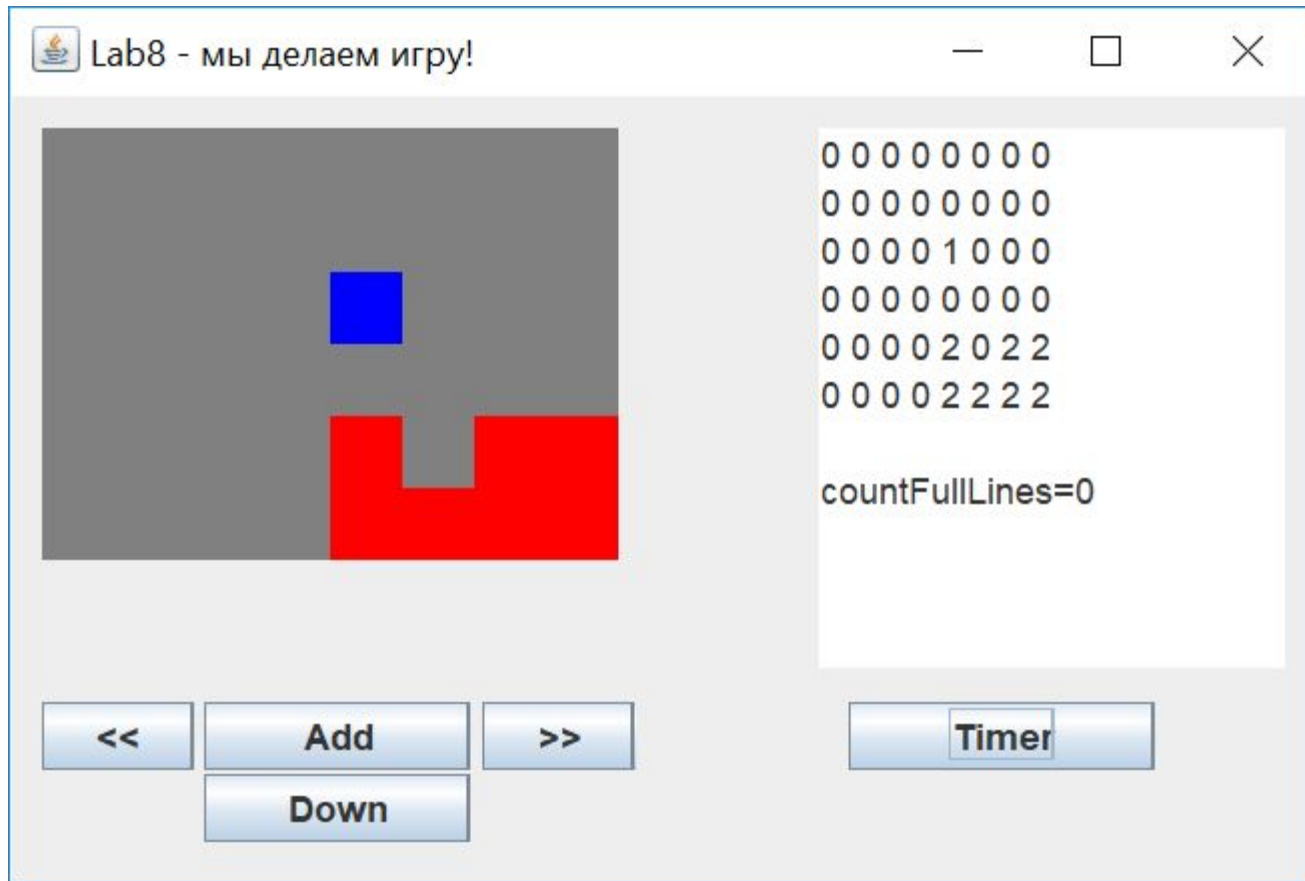
Власенко Олег Федосович

Лекция 4
Знакомство с таймером, клавиатурой и мышкой

Создаем заготовку для игры



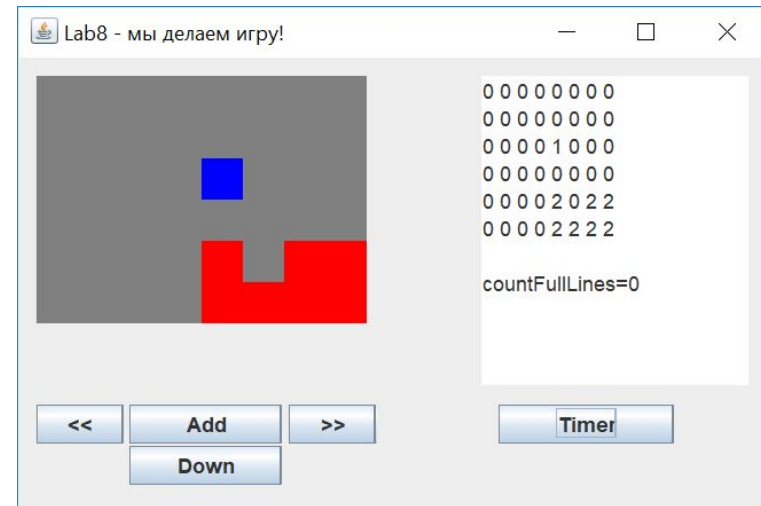
Создаем заготовку для игры



Создаем заготовку для игры

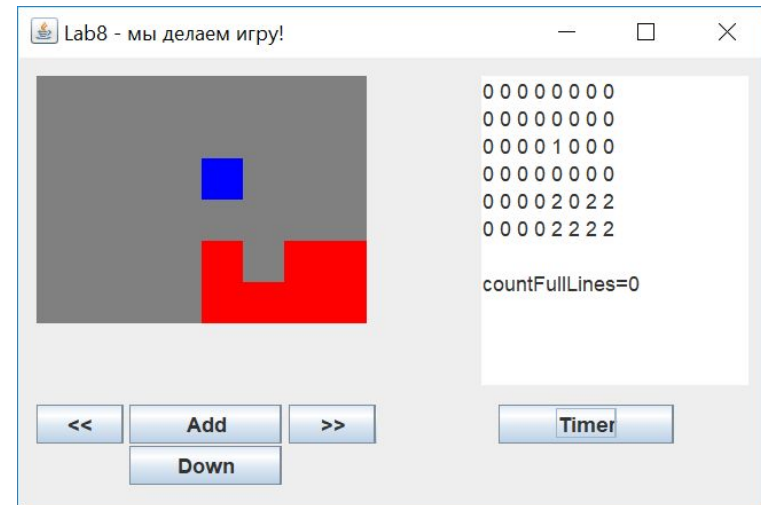
```
btnAddFigure = new JButton("Add");
btnAddFigure.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0) {
        tetrisArray.addFigure();
        textArea.setText(tetrisArray.toString());
        panel.repaint();

        if (tetrisArray.isGameOver()) {
            btnToLeft.setEnabled(false);
            btnAddFigure.setEnabled(false);
            btnToRight.setEnabled(false);
            btnDown.setEnabled(false);
            btnTimer.setEnabled(false);
        }
    }
});
btnAddFigure.setBounds(64, 202, 89, 23);
frmLab.getContentPane().add(btnAddFigure);
```



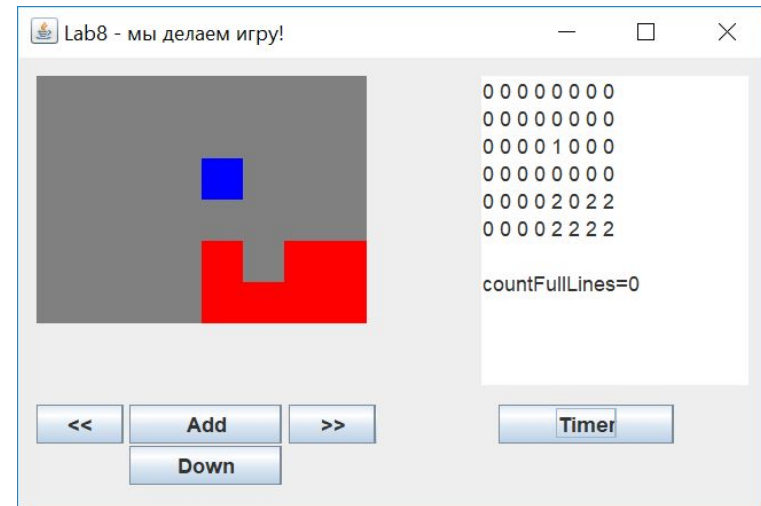
Создаем заготовку для игры

```
btnToLeft = new JButton("<<");
btnToLeft.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        tetrisArray.toLeft();
        textArea.setText(tetrisArray.toString());
        panel.repaint();
    }
});
btnToLeft.setBounds(10, 202, 51, 23);
frmLab.getContentPane().add(btnToLeft);
```



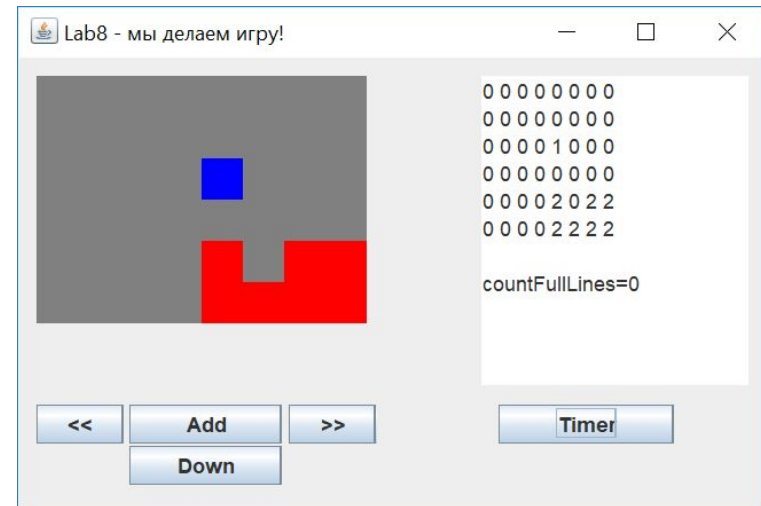
Создаем заготовку для игры

```
btnToRight = new JButton(">>");  
btnToRight.addActionListener(new ActionListener() {  
    public void actionPerformed(ActionEvent e) {  
        tetrisArray.toRight();  
        textArea.setText(tetrisArray.toString());  
        panel.repaint();  
    }  
});  
btnToRight.setBounds(157, 202, 51, 23);  
frmLab.getContentPane().add(btnToRight);
```



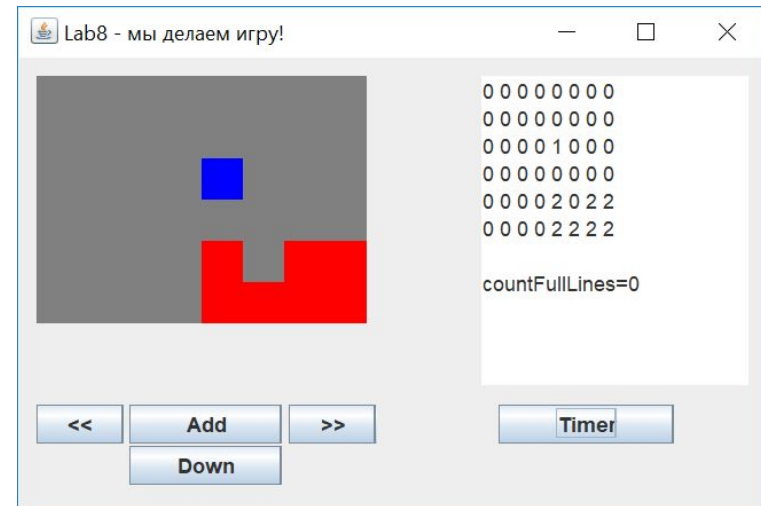
Создаем заготовку для игры

```
btnDown = new JButton("Down");  
btnDown.addActionListener(new ActionListener() {  
    public void actionPerformed(ActionEvent e) {  
        tetrisArray.fallToBottom();  
        textArea.setText(tetrisArray.toString());  
        panel.repaint();  
    }  
});  
btnDown.setBounds(64, 226, 89, 23);  
frmLab.getContentPane().add(btnDown);
```



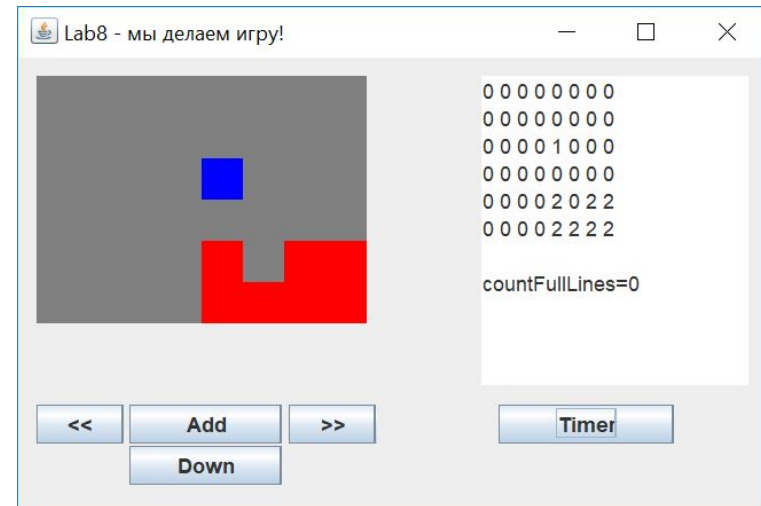
Создаем заготовку для игры

```
btnTimer = new JButton("Timer");  
btnTimer.addActionListener(new ActionListener() {  
    public void actionPerformed(ActionEvent arg0) {  
        tetrisArray.fallOneLine();  
        textArea.setText(tetrisArray.toString());  
        panel.repaint();  
    }  
});  
btnTimer.setBounds(279, 202, 102, 23);  
frmLab.getContentPane().add(btnTimer);
```



Создаем заготовку для игры – TetrisArray

```
public class TetrisArray {  
  
    private int [][]array;  
    private int lastI;  
    private int lastJ;  
  
    public TetrisArray() {  
        array = new int[6][8];  
        lastI = array.length - 1;  
        lastJ = array[lastI].length - 1;  
    }  
  
    private int countFullLines = 0;  
    private boolean gamelsOver = false;
```



Создаем заготовку для игры – TetrisArray

```
public int getGameResult() {  
    return countFullLines;  
}
```

```
public void addFigure() {  
    if (canAddFigure()) {  
        array[0][3] = 1;  
    } else {  
        gamelsOver = true;  
    }  
}
```

```
public void addBrick(int i, int j) {  
    if (array[i][j] == 0) {  
        array[i][j] = 2;  
    }  
}
```

Создаем заготовку для игры – TetrisArray

```
public void deleteBrick(int i, int j) {  
    if (array[i][j] == 2) {  
        array[i][j] = 0;  
    }  
}
```

```
public boolean isGameOver() {  
    return gamelsOver;  
}
```

```
boolean canAddFigure() {  
    if (array[0][3] == 0) {  
        return true;  
    } else {  
        return false;  
    }  
}
```

Создаем заготовку для игры – TetrisArray

```
public void toLeft() {  
    if (!figureExist()) {  
        return;  
    }  
    if (canMoveToLeft()) {  
        moveToLeft();  
    } else {  
        fastenFigure();  
    }  
}
```

```
public void toRight() {  
    if (!figureExist()) {  
        return;  
    }  
    if (canMoveToRight()) {  
        moveToRight();  
    } else {  
        fastenFigure();  
    }  
}
```

Создаем заготовку для игры – TetrisArray

```
void moveToLeft() {  
    for (int i = 0; i <= lastI; i++) {  
        for (int j = 1; j <= lastJ; j++) {  
            if (array[i][j] == 1) {  
                array[i][j - 1] = 1;  
                array[i][j] = 0;  
            }  
        }  
    }  
}
```

```
void moveToRight() {  
    for (int i = 0; i < lastI; i++) {  
        for (int j = lastJ - 1; j >= 0; j--) {  
            ...  
        }  
    }  
}
```

Создаем заготовку для игры – TetrisArray

```
public void fallToBottom() {  
    if (!figureExist()) {  
        return;  
    }  
    while (canMoveDown()) {  
        moveDown();  
    }  
    fastenFigure();  
}
```

```
public void fallOneLine() {  
    if (!figureExist()) {  
        return;  
    }  
    if (canMoveDown()) {  
        moveDown();  
    } else {  
        fastenFigure();  
    }  
}
```

Создаем заготовку для игры – TetrisArray

```
void moveDown() {  
    // Начинаем с предпоследней строки  
    for (int i = lastI - 1; i >= 0; i--) {  
        for (int j = 0; j <= lastJ; j++) {  
            if (array[i][j] == 1) {  
                array[i + 1][j] = 1;  
                array[i][j] = 0;  
            }  
        }  
    }  
}
```

Создаем заготовку для игры – TetrisArray

```
boolean bottomLineIsFull() {  
    for (int j = 0; j <= lastJ; j++) {  
        if (array[lastI][j] != 2) {  
            return false;  
        }  
    }  
    return true;  
}
```


Создаем заготовку для игры – TetrisArray

```
void deleteBottomLine() {  
    for (int i = lastI - 1; i >= 0; i--) {  
        for (int j = 0; j <= lastJ; j++) {  
            array[i + 1][j] = array[i][j];  
        }  
    }  
    for (int j = 0; j <= lastJ; j++) {  
        array[0][j] = 0;  
    }  
}
```

Создаем заготовку для игры – TetrisArray

```
public boolean figureExist() {  
    for (int i = 0; i <= lastI; i++) {  
        for (int j = 0; j <= lastJ; j++) {  
            if (array[i][j] == 1) {  
                return true;  
            }  
        }  
    }  
    return false;  
}
```

Создаем заготовку для игры – TetrisArray

```
boolean canMoveDown() {  
    for (int i = 0; i < lastI; i++) {  
        for (int j = 0; j <= lastJ; j++) {  
            if (array[i][j] == 1) {  
                if (array[i + 1][j] == 2) {  
                    return false;  
                }  
            }  
        }  
    }  
    for (int j = 0; j <= lastJ; j++) {  
        if (array[lastI][j] == 1) {  
            return false;  
        }  
    }  
    return true;  
}
```

Создаем заготовку для игры – TetrisArray

```
boolean canMoveToLeft() {  
    for (int i = 0; i <= lastI; i++) {  
        for (int j = 1; j <= lastJ; j++) {  
            if (array[i][j] == 1) {  
                if (array[i][j - 1] == 2) {  
                    return false;  
                }  
            }  
        }  
    }  
    for (int i = 0; i <= lastI; i++) {  
        if (array[i][0] == 1) {  
            return false;  
        }  
    }  
    return true;  
}
```

Создаем заготовку для игры – TetrisArray

```
boolean canMoveToRight() {  
    for (int i = 0; i <= lastI; i++) {  
        for (int j = 0; j < lastJ; j++) {  
            if (array[i][j] == 1) {  
                if (array[i][j + 1] == 2) {  
                    return false;  
                }  
            }  
        }  
    }  
    for (int i = 0; i <= lastI; i++) {  
        if (array[i][lastJ] == 1) {  
            return false;  
        }  
    }  
    return true;  
}
```

Создаем заготовку для игры – TetrisArray

```
void fastenFigure() {  
    for (int i = 0; i <= lastI; i++) {  
        for (int j = 0; j <= lastJ; j++) {  
            if (array[i][j] == 1) {  
                array[i][j] = 2;  
            }  
        }  
    }  
    if (bottomLineIsFull()) {  
        deleteBottomLine();  
        countFullLines++;  
    }  
}
```

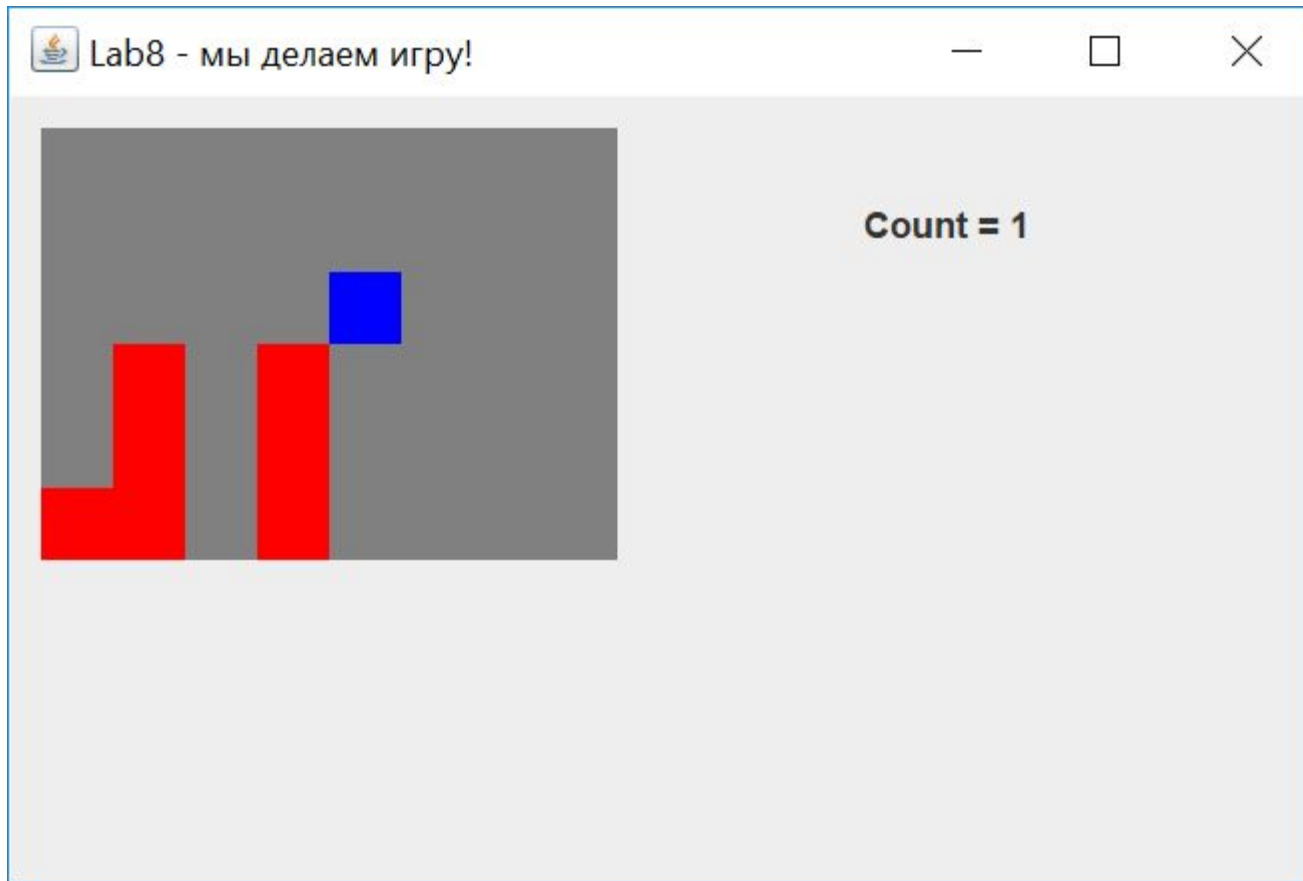
Создаем заготовку для игры – TetrisArray

```
public String toString() {  
    String str = "";  
    for (int i = 0; i <= lastI; i++) {  
        for (int j = 0; j <= lastJ; j++) {  
            str = str + array[i][j] + " ";  
        }  
        str = str + "\n";  
    }  
    str = str + "\n";  
    str = str + "countFullLines=" + countFullLines + "\n";  
    if (gameIsOver) {  
        str = str + "GAME IS OVER!!!" + "\n";  
    }  
    return str;  
}
```

Создаем заготовку для игры – TetrisArray

```
final public static int CELL_HEIGHT = 24;
final public static int CELL_WIDTH = 24;
public void drawArray(Graphics g, int width, int height) {
    for (int i = 0; i < array.length; i++) {
        int top = i * CELL_HEIGHT;
        for (int j = 0; j < array[i].length; j++) {
            int left = j * CELL_WIDTH;
            if (array[i][j] == 0) {
                g.setColor(Color.GRAY);
            } else if (array[i][j] == 1) {
                g.setColor(Color.BLUE);
            } else if (array[i][j] == 2) {
                g.setColor(Color.RED);
            }
            g.fillRect(left, top, CELL_WIDTH, CELL_HEIGHT);
        }
    }
}
```


Создаем почти игру



Создаем почти игру - interface Updated

```
public interface Updated {  
    void update();  
}
```

Создаем почти игру - Win_TetrisGame

```
public class Win_TetrisGame  
implements KeyListener, Updated {
```

```
...
```

```
TetrisArray tetrisArray;  
private JPanel panel;  
private javax.swing.Timer timer;  
private JLabel lblGameResult;  
private JLabel lblGameOver;
```

Создаем почти игру - Win_TetrisGame

```
/**
 * Initialize the contents of the frame.
 */
private void initialize() {
    frmLab = new JFrame();
    frmLab.setTitle("Lab8 - \u043C\u044B\u0434\u0435\u043B\u0430 \u0438\u0433\u0440\u044B!");
    frmLab.setBounds(100, 100, 450, 300);
    frmLab.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frmLab.getContentPane().setLayout(null);
    frmLab.addKeyListener(this); // Форма СЛУШАЕТ клавиатуру
    tetrisArray = new TetrisArray();
    panel = new TetrisPanel(tetrisArray, this);
    panel.setBounds(10, 11, 249, 180);
    frmLab.getContentPane().add(panel);
}
```

Создаем почти игру - Win_TetrisGame

```
lblGameResult = new JLabel("Count = 0");  
lblGameResult.setBounds(284, 35, 113, 14);  
frmLab.getContentPane().add(lblGameResult);
```

```
lblGameOver = new JLabel("");  
lblGameOver.setFont(new Font("Traditional Arabic", Font.BOLD, 14));  
lblGameOver.setBounds(277, 84, 131, 14);  
frmLab.getContentPane().add(lblGameOver);
```

Создаем почти игру - Win_TetrisGame

//Будет вызываться каждые 1000 мсек

```
timer = new Timer(1000, new ActionListener() {  
    public void actionPerformed(ActionEvent e) {  
        if (tetrisArray.figureExist()) {  
            tetrisArray.fallOneLine();  
        } else {  
            tetrisArray.addFigure();  
        }  
        panel.repaint();  
        String res = "Count = " + tetrisArray.getGameResult();  
        lblGameResult.setText(res);  
        if (tetrisArray.isGameOver()) {  
            lblGameOver.setText("GAME OVER!");  
        }  
    }  
});  
timer.start();
```

Создаем почти игру - Win_TetrisGame

```
}
```

```
@Override
```

```
public void keyReleased(KeyEvent arg0) {
```

```
    switch(arg0.getKeyCode()) {
```

```
    case KeyEvent.VK_LEFT:
```

```
        tetrisArray.toLeft();
```

```
        panel.repaint();
```

```
        break;
```

```
    case KeyEvent.VK_RIGHT:
```

```
        tetrisArray.toRight();
```

```
        panel.repaint();
```

```
        break;
```

Создаем почти игру - Win_TetrisGame

```
case KeyEvent.VK_DOWN:
    tetrisArray.fallToBottom();
    panel.repaint();
    break;
default:
    return;
}
```

```
String res = "Count = " + tetrisArray.getGameResult();
lblGameResult.setText(res);
```

```
if (tetrisArray.isGameOver()) {
    lblGameOver.setText("GAME OVER!");
}
```

```
}
```


Создаем почти игру - Win_TetrisGame

```
@Override
```

```
public void keyPressed(KeyEvent arg0) {  
    // TODO Auto-generated method stub  
}
```

```
@Override
```

```
public void keyTyped(KeyEvent e) {  
    // TODO Auto-generated method stub  
}
```

```
@Override
```

```
public void update() {  
    panel.repaint();  
}
```

```
}
```

Создаем почти игру - TetrisPanel

```
import java.awt.Graphics;  
import java.awt.event.MouseEvent;  
import java.awt.event.MouseListener;  
import javax.swing.JPanel;
```

```
public class TetrisPanel extends JPanel implements MouseListener {
```

```
    TetrisArray arr;
```

```
    Updated win;
```

```
    public TetrisPanel(TetrisArray array, Updated window) {
```

```
        arr = array;
```

```
        win = window;
```

```
        addMouseListener(this); // Слушатель ТЕПЕРЬ слушает!!!
```

```
    }
```

Создаем почти игру - TetrisPanel

```
public void paint(Graphics g) {  
    super.paint(g);  
    arr.drawArray(g, this.getWidth(), this.getHeight());  
}
```

Создаем почти игру - TetrisPanel

@Override

```
public void mouseClicked(MouseEvent me) {  
    int i = me.getY() / TetrisArray.CELL_HEIGHT;  
    int j = me.getX() / TetrisArray.CELL_WIDTH;  
    int cntClicks = me.getClickCount();  
  
    if (cntClicks == 2) {  
        arr.addBrick(i, j);  
    } else {  
        arr.deleteBrick(i, j);  
    }  
    //repaint();  
    win.update();  
}
```

Создаем почти игру - TetrisPanel

```
@Override
```

```
public void mouseEntered(MouseEvent arg0) {  
    // TODO Auto-generated method stub  
}
```

```
@Override
```

```
public void mouseExited(MouseEvent arg0) {  
    // TODO Auto-generated method stub  
}
```

Создаем почти игру - TetrisPanel

```
@Override
```

```
public void mousePressed(MouseEvent arg0) {
```

```
    // TODO Auto-generated method stub
```

```
}
```

```
@Override
```

```
public void mouseReleased(MouseEvent arg0) {
```

```
    // TODO Auto-generated method stub
```

```
}
```

```
}
```

Создаем почти игру

MouseListener, правая и левая кнопка мышиMouseListener,
правая и левая кнопка мыши - <https://javatalks.ru/topics/4964>

indeed

Как различить эти две кнопки? Они абсолютно равноценны для вызова какого либо события. Или в java это сделать невозможно?

Староверъ

- 1.Отслеживаете событие MouseEvent.
А потом
проверяете:if(ev.getButton()==MouseEvent.BUTTON1)
Это левая кнопка. Правая - Button3.

Спасибо за внимание!

Власенко Олег Федосович

E-mail: vlasenko.oleg@gmail.com

Vk: vk.com/oleg.f.vlasenko

Телефон: 8 902 246 05 47