Tetris.java

```
package pat.tetris;

import javax.swing.JFrame;

public class Tetris extends JFrame {

    public Tetris() {
        initUI();
    }

    private void initUI() {
        add(new Board());
        setTitle("Tetris");
        setSize(200, 400);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setLocationRelativeTo(null);
    }

    public static void main(String[] args) {
        Tetris game = new Tetris();
        game.setVisible(true);
    }
}
```

Board.java

```
package pat.tetris;

import java.awt.Color;
import java.awt.Dimension;
import java.awt.Graphics;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.KeyAdapter;
import java.awt.event.KeyEvent;
import javax.swing.JPanel;
import javax.swing.Timer;

public class Board extends JPanel implements ActionListener {

    private final int BOARD_WIDTH = 10;
    private final int BOARD_HEIGHT = 20;
    private Timer timer;
    private boolean isFallingFinished = false;
    private boolean isStarted = false;
    private boolean isPaused = false;

    public Board() {
        initBoard();
        addKeyListener(new KeyAdapter() {
            @Override
            public void keyPressed(KeyEvent e) {
                int keycode = e.getKeyCode();
                switch (keycode) {
                    case KeyEvent.VK_LEFT:
                        movePieceLeft();
                        break;
                    case KeyEvent.VK_RIGHT:
                        movePieceRight();
                        break;
                    case KeyEvent.VK_DOWN:
                        dropPiece();
                        break;
                    case KeyEvent.VK_UP:
                        rotatePiece();
                        break;
                }
            }
        });
    }

    private void initBoard() {
        setFocusable(true);
        setBackground(Color.BLACK);
        setPreferredSize(new Dimension(200, 400));
        timer = new Timer(400, this); // Le timer pour le mouvement des pièces
        timer.start();
```

```
    }

    @Override
    public void actionPerformed(ActionEvent e) {
        // La logique de jeu est exécutée ici à chaque tick du timer
        if (isFallingFinished) {
            isFallingFinished = false;
            // Créer une nouvelle pièce et la placer en haut
        }
        // Logique pour déplacer les pièces vers le bas
        repaint();
    }

    @Override
    public void paintComponent(Graphics g) {
        super.paintComponent(g);
        // Logique pour dessiner le jeu
    }

    private void movePieceLeft() {
        // Logique pour déplacer la pièce vers la gauche
    }

    private void movePieceRight() {
        // Logique pour déplacer la pièce vers la droite
    }

    private void dropPiece() {
        // Logique pour accélérer la descente de la pièce
    }

    private void rotatePiece() {
        // Logique pour tourner la pièce
    }
}
```

Shape.java

```
package pat.tetris;

public class Shape {

    public enum Tetrominoe { NoShape, ZShape, SShape, LineShape, TShape, SquareShape,
LShape, MirroredLShape }

    private Tetrominoe pieceShape;
    private int[][] coords;

private void setCoordinates(Tetrominoe shape) {
    switch (shape) {
        case ZShape:
            coords = new int[][] { {0, -1}, {0, 0}, {1, 0}, {1, 1} };
            break;
        // Définir les autres formes...
    }
}

    public Shape() {
        coords = new int[4][2];
        setShape(Tetrominoe.NoShape);
    }

    public void setShape(Tetrominoe shape) {
        // Définir les coordonnées de la forme en fonction du type de pièce
    }

    public void setRandomShape() {
        // Sélectionner une forme aléatoire
    }

    public int getX(int index) { return coords[index][0]; }
    public int getY(int index) { return coords[index][1]; }
    public Tetrominoe getShape() { return pieceShape; }
}
```