```java
// TrisGame.java
import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.Container;
import java.awt.FlowLayout;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.BorderFactory;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.JPanel;

public class TrisGame extends JFrame {

    private CellaTris[][] celle;
    private MyLabel lblGiocatoreX, lblGiocatoreO;
    private JButton btnNuovaPartita;
    private char turnoAttuale;
    private boolean partitaTerminata;

    public TrisGame() {
        super("Gioco del Tris");

        // Inizializzazione componenti
        celle = new CellaTris[3][3];
        turnoAttuale = 'X';
        partitaTerminata = false;

        this.setDefaultCloseOperation(EXIT_ON_CLOSE);
        this.setLocationRelativeTo(null);

        initComponenti();
        initPannelli();
        initAscoltatori();

        pack();
        setVisible(true);
        btnNuovaPartita.doClick();
    }

    private void initComponenti() {
        // Inizializzazione delle celle
        for(int i = 0; i < 3; i++) {
            for(int j = 0; j < 3; j++) {
                celle[i][j] = new CellaTris(i, j);
```

```
            }
        }

        // Etichette per il punteggio
        lblGiocatoreX = new MyLabel("Giocatore X", 0);
        lblGiocatoreO = new MyLabel("Giocatore O", 0);

        // Pulsante nuova partita
        btnNuovaPartita = new JButton("Nuova Partita");
    }

    private void initPannelli() {
        // Pannello superiore con punteggio
        JPanel pnlNord = new JPanel(new FlowLayout());
        pnlNord.add(lblGiocatoreX);
        pnlNord.add(lblGiocatoreO);

        // Pannello centrale con griglia di celle
        JPanel pnlCentro = new JPanel(new GridLayout(3, 3, 5, 5));
        pnlCentro.setBackground(Color.GRAY);
        pnlCentro.setBorder(BorderFactory.createEmptyBorder(10, 10, 10,
10));

        for(int i = 0; i < 3; i++) {
            for(int j = 0; j < 3; j++) {
                pnlCentro.add(celle[i][j]);
            }
        }

        // Pannello inferiore con pulsante nuova partita
        JPanel pnlSud = new JPanel(new FlowLayout(FlowLayout.CENTER));
        pnlSud.add(btnNuovaPartita);

        // Aggiunta pannelli al contenitore principale
        Container contenitore = this.getContentPane();
        contenitore.add(pnlNord, BorderLayout.NORTH);
        contenitore.add(pnlCentro, BorderLayout.CENTER);
        contenitore.add(pnlSud, BorderLayout.SOUTH);
    }

    private void initAscoltatori() {
        // Ascoltatore con classe esterna per le celle
        AscoltaCella ascoltaCella = new AscoltaCella(this);

        for(int i = 0; i < 3; i++) {
            for(int j = 0; j < 3; j++) {
                celle[i][j].addActionListener(ascoltaCella);
            }
        }
```

```
        // Ascoltatore con classe anonima per il pulsante nuova partita
        btnNuovaPartita.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                // Reset della griglia
                for(int i = 0; i < 3; i++) {
                    for(int j = 0; j < 3; j++) {
                        celle[i][j].resetCella();
                    }
                }
                turnoAttuale = 'X';
                partitaTerminata = false;
            }
        });
    }

    // Metodo per cambiare turno (utilizzo di classe interna)
    public void cambioTurno() {
        GestoreTurno gestore = new GestoreTurno();
        gestore.cambiaTurno();
    }

    // Metodo per controllare vincita
    public boolean controllaVincita() {
        // Controllo righe
        for(int i = 0; i < 3; i++) {
            if(celle[i][0].getValore() != ' ' &&
               celle[i][0].getValore() == celle[i][1].getValore() &&
               celle[i][1].getValore() == celle[i][2].getValore()) {
                return true;
            }
        }

        // Controllo colonne
        for(int j = 0; j < 3; j++) {
            if(celle[0][j].getValore() != ' ' &&
               celle[0][j].getValore() == celle[1][j].getValore() &&
               celle[1][j].getValore() == celle[2][j].getValore()) {
                return true;
            }
        }

        // Controllo diagonali
        if(celle[0][0].getValore() != ' ' &&
           celle[0][0].getValore() == celle[1][1].getValore() &&
           celle[1][1].getValore() == celle[2][2].getValore()) {
            return true;
        }

        if(celle[0][2].getValore() != ' ' &&
```

```
            celle[0][2].getValore() == celle[1][1].getValore() &&
            celle[1][1].getValore() == celle[2][0].getValore()) {
             return true;
        }

        return false;
    }

    // Metodo per controllare pareggio
    public boolean controllaPareggio() {
        for(int i = 0; i < 3; i++) {
            for(int j = 0; j < 3; j++) {
                if(celle[i][j].getValore() == ' ') {
                    return false;
                }
            }
        }
        return true;
    }

    // Getter e setter
    public char getTurnoAttuale() {
        return turnoAttuale;
    }

    public boolean isPartitaTerminata() {
        return partitaTerminata;
    }

    public void setPartitaTerminata(boolean partitaTerminata) {
        this.partitaTerminata = partitaTerminata;
    }

    public void incrementaPunteggioX() {
        lblGiocatoreX.incrementa();
    }

    public void incrementaPunteggioO() {
        lblGiocatoreO.incrementa();
    }

    // Classe interna per gestire il cambio turno
    private class GestoreTurno {
        public void cambiaTurno() {
            turnoAttuale = (turnoAttuale == 'X') ? 'O' : 'X';
        }
    }

    // Main per avvio applicazione
    public static void main(String[] args) {
```

```
        new TrisGame();
    }
}

// CellaTris.java
import java.awt.Color;
import java.awt.Dimension;
import java.awt.Font;
import javax.swing.JButton;

public class CellaTris extends JButton {

    private int riga;
    private int colonna;
    private char valore;

    public CellaTris(int riga, int colonna) {
        super();
        this.riga = riga;
        this.colonna = colonna;
        this.valore = ' ';

        setFont(new Font("Arial", Font.BOLD, 40));
        setBackground(Color.WHITE);
        setPreferredSize(new Dimension(80, 80));
        setText("");
    }

    public void setValore(char valore) {
        this.valore = valore;
        setText(String.valueOf(valore));

        // Cambio colore in base al valore
        if(valore == 'X') {
            setForeground(Color.BLUE);
        } else if(valore == 'O') {
            setForeground(Color.RED);
        }
    }

    public char getValore() {
        return valore;
    }

    public void resetCella() {
        valore = ' ';
        setText("");
        setEnabled(true);
    }
```

```
    public int getRiga() {
        return riga;
    }

    public int getColonna() {
        return colonna;
    }
}

// AscoltaCella.java
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.JOptionPane;

public class AscoltaCella implements ActionListener {

    private TrisGame gioco;

    public AscoltaCella(TrisGame gioco) {
        this.gioco = gioco;
    }

    @Override
    public void actionPerformed(ActionEvent e) {
        // Se la partita è terminata, non fare nulla
        if(gioco.isPartitaTerminata()) {
            return;
        }

        // Ottenere la cella cliccata
        CellaTris cella = (CellaTris) e.getSource();

        // Se la cella è già occupata, non fare nulla
        if(cella.getValore() != ' ') {
            return;
        }

        // Imposta il valore della cella in base al turno
        cella.setValore(gioco.getTurnoAttuale());
        cella.setEnabled(false);

        // Controlla vincita
        if(gioco.controllaVincita()) {
            gioco.setPartitaTerminata(true);

            // Aggiorna punteggio
            if(gioco.getTurnoAttuale() == 'X') {
                gioco.incrementaPunteggioX();
            } else {
                gioco.incrementaPunteggioO();
```

```
            }

            // Mostra messaggio
            JOptionPane.showMessageDialog(gioco,
                "Il giocatore " + gioco.getTurnoAttuale() + " ha vinto!",
                "Vittoria",
                JOptionPane.INFORMATION_MESSAGE);

            return;
        }

        // Controlla pareggio
        if(gioco.controllaPareggio()) {
            gioco.setPartitaTerminata(true);
            JOptionPane.showMessageDialog(gioco,
                "La partita è terminata in pareggio!",
                "Pareggio",
                JOptionPane.INFORMATION_MESSAGE);
            return;
        }

        // Cambia turno
        gioco.cambioTurno();
    }
}

// MyLabel.java
import java.awt.Font;
import javax.swing.JLabel;

public class MyLabel extends JLabel {

    private String etichetta;
    private int valore;

    public MyLabel(String etichetta, int valore) {
        super();
        this.etichetta = etichetta + " ";
        this.valore = valore;
        this.setHorizontalAlignment(JLabel.CENTER);
        setFont(new Font("sans-serif", Font.PLAIN, 20));
        setText(this.etichetta + valore);
    }

    public void setValore(int valore) {
        this.valore = valore;
        setText(etichetta + valore);
    }

    public int getValore() {
```

```
        return valore;
    }

    public void incrementa() {
        setValore(++valore);
    }

    public void decrementa() {
        setValore(--valore);
    }

    public void aggiungi(int valore) {
        this.valore += valore;
        setText(etichetta + this.valore);
    }
}
```