```java
package example;
import java.awt.BorderLayout;

/**
 * @author Mike Harris & Scott Palmer
 * @version 1.0
 *
 * Description:
 * RaceGame.java creates a simple GUI of a race track. Two players can play
where Player 1 is A and
 * Player 2 is B. Either player starts on opposite end of the track. Player 1
starts first with a roll.
 * The player then can choose a number and re-roll that die or choose the other
one.  After making two
 * choices of spaces to move the turn changes for Player 2. If players collide,
the pieces share the
 * space and play continues. The winner is announced when a player reaches
 * the opposite end of the track.
 *
 */

public class RaceGame implements ActionListener {
    static final int NO_PLAYER = 0;
    static final int PLAYER1 = 1;
    static final int PLAYER2 = 2;

    static final int PLAYER1_START = 0;
    static final int PLAYER2_START = 14;
    static final int TOTAL_POSITIONS = 15;

    final String NO_NUM = "-";

    // Interface Components
    JButton leftDie1;
    JButton leftDie2;
    JButton rollButton;
    JButton rightDie1;
    JButton rightDie2;
    JLabel[] trackPosition;
    JPanel raceTrack;
    JLabel mesg;

    // initialize starting positions for game
    boolean roll = true;
    int player1Pos = PLAYER1_START;
```

```
    int player2Pos = PLAYER2_START;

    int winner = NO_PLAYER;

    int numClicks = 0;
    int turnID = 1;

    public static void main(String[] args) {
        // Schedule App's GUI create & show for event-dispatching thread
        javax.swing.SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                createAndShowGUI();
            }
        });
    }

     /**
      * Creates the GUI and shows it. Invoked from the event-dispatching thread
for
      * thread safety.
      * @param
      */
    private static void createAndShowGUI() {

        // Create and set up the window.
        JFrame frame = new JFrame("Race Game");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        Container cont = frame.getContentPane();

        RaceGame app = new RaceGame();
        app.init(cont);

        // Display the window.
        frame.pack();
        frame.setVisible(true);
    }

    /**
     * Initializes button layout and board view.
      * @param cont
      */
     public void init(Container cont) {

         // initialize button label prior to roll
        leftDie1 = new JButton(NO_NUM);
```

```
        leftDie1.addActionListener(this);
        leftDie2 = new JButton(NO_NUM);
        leftDie2.addActionListener(this);

        rightDie1 = new JButton(NO_NUM);
        rightDie1.addActionListener(this);
        rightDie2 = new JButton(NO_NUM);
        rightDie2.addActionListener(this);

        rollButton = new JButton("Roll");
        rollButton.addActionListener(this);

        cont.setLayout(new BorderLayout());
        JPanel buttonRow = new JPanel();
        buttonRow.setLayout(new GridLayout(1, 0));

        // Add the buttons to form the player command row
        buttonRow.add(leftDie1);
        buttonRow.add(leftDie2);
        buttonRow.add(rollButton);
        buttonRow.add(rightDie1);
        buttonRow.add(rightDie2);

        // Setting up game board for start of play
        raceTrack = new JPanel(new GridLayout(3, 5));

        raceTrack.setPreferredSize(new Dimension(320, 175));
        raceTrack.setBackground(Color.LIGHT_GRAY);
        trackPosition = new JLabel[TOTAL_POSITIONS];
        for (int i = 0; i < TOTAL_POSITIONS; i++) {
            trackPosition[i] = new JLabel(i + ":", JLabel.CENTER);
            trackPosition[i].setBorder(BorderFactory
                    .createLineBorder(Color.black));
            raceTrack.add(trackPosition[i]);
        }
        trackPosition[PLAYER1_START].setText("0:A");
        trackPosition[PLAYER2_START].setText("14:B");
        cont.add(raceTrack, BorderLayout.CENTER);
        cont.add(buttonRow, BorderLayout.SOUTH);

        mesg = new JLabel("Player 1, Press Roll To Start Race Game");
        cont.add(mesg, BorderLayout.NORTH);
    }

    /**
```

```
     * Invoked when an action occurs.
     * @param ae
     */
    public void actionPerformed(ActionEvent ae) {
        // Find which button was pressed
        JButton actionButton = (JButton) ae.getSource();

        if (actionButton.getText().contains(NO_NUM)
                || actionButton.getText().contains(NO_NUM))
            return;

        buttonCheck(actionButton);

        // Determine if anyone has won yet
        if (player1Pos >= PLAYER2_START)
            winner = PLAYER1;
        else if (player2Pos <= 0)
            winner = PLAYER2;

        // Determine if time to roll again
        if (numClicks == 2) {
            if (turnID == 1) {
                turnID = 2;
                roll = true;
                rightDie1.setText(NO_NUM);
                rightDie2.setText(NO_NUM);
            } else {
                turnID = 1;
                roll = true;
                leftDie1.setText(NO_NUM);
                leftDie2.setText(NO_NUM);
            }

            numClicks = 0;
            if (winner == NO_PLAYER)
                mesg.setText("Player " + turnID + ", Time to hit Roll again");
            else if (winner == PLAYER1)
                mesg.setText("Player 1 Wins!");
            else
                mesg.setText("player 2 Wins!");
        }

    }

    /**
```

```
    * Evaluates the event action for all buttons and updates the game board
    * NOTE dice rolling is hard-coded, change to use random
    * @param buttonClicked
    */
   public void buttonCheck(JButton buttonClicked) {

      int rollValue;

       // retrieve value of selected die
       if (buttonClicked != rollButton) {
           rollValue = Integer.valueOf(buttonClicked.getText());
           buttonClicked.setText(NO_NUM);
           numClicks++;

           // update game board with new player position
           // Player1
           if ((buttonClicked == leftDie1) || (buttonClicked == leftDie2)) {
               // Remove player1 piece from track
               if (player1Pos == player2Pos)
                   trackPosition[player1Pos].setText(player1Pos + ":B");
               else
                   trackPosition[player1Pos].setText(player1Pos + ":");

               // Apply roll value to player1 position
               if (player1Pos < PLAYER2_START)
                   player1Pos += rollValue;

               // Place player1 on track
               if (player1Pos == player2Pos)
                   trackPosition[player1Pos].setText(player1Pos + ":AB");
               else
                   trackPosition[player1Pos].setText(player1Pos + ":A");
           }
           // Player2
           else if ((buttonClicked == rightDie1) || (buttonClicked ==
rightDie2)) {
               // Remove player2 piece from track
               if (player1Pos == player2Pos)
                   trackPosition[player2Pos].setText(player2Pos + ":A");
               else
                   trackPosition[player2Pos].setText(player2Pos + ":");

               // Apply roll value to player2 position
               if (player2Pos > PLAYER1_START)
                   player2Pos -= rollValue;
```

```
                // Place player2 on track
                if (player1Pos == player2Pos)
                    trackPosition[player2Pos].setText(player2Pos + ":AB");
                else
                    trackPosition[player2Pos].setText(player2Pos + ":B");
            }
        }

        // or generate new values
        else {
            if (turnID == 1) {
                leftDie1.setText("1");
                leftDie2.setText("4");
                mesg.setText("Player 1, click a number on the left");
                roll = false;
            } else {
                rightDie1.setText("4");
                rightDie2.setText("1");
                mesg.setText("Player 2, click a number on the right");
                roll = false;
            }

            return;
        }
    }
}
```