

# ConnectFour.java

```
/**
 * The ConnectFour class.
 *
 * This class represents a Connect Four (TM)
 * game, which allows two players to drop
 * checkers into a grid until one achieves
 * four checkers in a straight line.
 */
public class ConnectFour {

    // the grid used for storing the game layout.
    private int[][] grid;
    // the player whose turn it is.
    private int currentPlayer;

    /**
     * The ConnectFour constructor.
     *
     * Creates and initializes the grid for the
     * Connect Four game.
     */
    public ConnectFour() {
        // create the grid
        grid = new int[7][6];
        // initialize the grid
        for (int row=0; row<6; row++) {
            for (int column=0; column<7; column++) {
                // set the position to a default value
                grid[column][row] = 0;
            }
        }
        // set the first move to Player 1
        currentPlayer = 1;
    }

    /**
     * The drop method.
     *
     * Drop a checker into the specified column,
     * and return the row that the checker lands on.
     */
    int drop(int column) {
        if (hasWon()) {
            return -1;
        }

        int row = 0;
        for ( ; row<6 && grid[column][row]!=0; row++) { };
        if (row==6) {
            // if the row is 6, it went through all 6 rows
            // of the grid, and couldn't find an empty one.
            // Therefore, return false to indicate that this
            // drop operation failed.
        }
    }
}
```

```

        return -1;
    }
    // fill the row of that column with a checker.
    grid[column][row] = currentPlayer;
    // alternate the players
    currentPlayer = (currentPlayer%2)+1;
    return row;
}

/**
 * The toString method
 *
 * Returns a String representation of this
 * Connect Four (TM) game.
 */
public String toString() {
    String returnString = "";
    for (int row=5; row>=0; row--) {
        for (int column=0; column<7; column++) {
            returnString = returnString + grid[column][row];
        }
        returnString = returnString + "\n";
    }
    return returnString;
}

/**
 * The hasWon method.
 *
 * This method returns true if one of the
 * players has won the game.
 */
public boolean hasWon() {
    boolean status = false;

    // check for a horizontal win
    for (int row=0; row<6; row++) {
        for (int column=0; column<4; column++) {
            if (grid[column][row] != 0 &&
                grid[column][row] == grid[column+1][row] &&
                grid[column][row] == grid[column+2][row] &&
                grid[column][row] == grid[column+3][row]) {
                status = true;
            }
        }
    }

    // check for a vertical win
    for (int row=0; row<3; row++) {
        for (int column=0; column<7; column++) {
            if (grid[column][row] != 0 &&
                grid[column][row] == grid[column][row+1] &&
                grid[column][row] == grid[column][row+2] &&
                grid[column][row] == grid[column][row+3]) {
                status = true;
            }
        }
    }
}

```

```

}

// check for a diagonal win (positive slope)
for (int row=0; row<3; row++) {
    for (int column=0; column<4; column++) {
        if (grid[column][row] != 0 &&
            grid[column][row] == grid[column+1][row+1] &&
            grid[column][row] == grid[column+2][row+2] &&
            grid[column][row] == grid[column+3][row+3]) {
            status = true;
        }
    }
}

// check for a diagonal win (negative slope)
for (int row=3; row<6; row++) {
    for (int column=0; column<4; column++) {
        if (grid[column][row] != 0 &&
            grid[column][row] == grid[column+1][row-1] &&
            grid[column][row] == grid[column+2][row-2] &&
            grid[column][row] == grid[column+3][row-3]) {
            status = true;
        }
    }
}

return status;
}
}

```

# ConnectFourGUI.java

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

```
public class ConnectFourGUI {
```

```
    private JFrame frame;
    private JLabel[][] slots;
    private int currentPlayer;
```

```
    public ConnectFourGUI() {
```

```
        frame = new JFrame("Connect Four");
        JPanel panel = (JPanel) frame.getContentPane();
        panel.setLayout(new GridLayout(6,7));
        slots = new JLabel[7][6];
        for (int row=5; row>=0; row--) {
            for (int column=0; column<7; column++) {
                slots[column][row] = new JLabel();
                // slots[column][row].setFont(new Font("SansSerif", Font.BOLD, 18));
```

set up the grid for  
the interface

```
                slots[column][row].setHorizontalAlignment(SwingConstants.CENTER);
                slots[column][row].setBorder(new LineBorder(Color.green));
                panel.add(slots[column][row]);
            }
        }
```

```
        frame.setContentPane(panel);
        frame.setSize(700,600);
        frame.setVisible(true);
```

display the  
interface

```
        currentPlayer = 1;
    }
```

```
    public void addListener(ConnectFourListener listener) {
```

```
        for (int row=0; row<6; row++) {
            for (int column=0; column<7; column++) {
                slots[column][row].addMouseListener(listener);
            }
        }
```

adds the listener to  
all of the JLabels

```
    }
```

```
    public int getColumn(JLabel label) {
```

```
        int returnColumn = -1;
        for (int row=0; row<6; row++) {
            for (int column=0; column<7; column++) {
                if (slots[column][row] == label) {
                    returnColumn = column;
                }
            }
        }
```

locate the column  
of a specified  
JLabel, so that the  
listener can invoke  
the drop method in  
the ConnectFour  
game

```
        return returnColumn;
    }
```

```
public void set(int column, int row) {
    // slots[column][row].setText("*" + currentPlayer + "*");
    if (currentPlayer == 1) {
        slots[column][row].setIcon(new ImageIcon("C:\\Documents and
Settings\\All Users\\Documents\\My Pictures\\Sample
Pictures\\Sunset.jpg"
));
    }
    else {
        slots[column][row].setIcon(new ImageIcon("C:\\Documents and
Settings\\All Users\\Documents\\My Pictures\\Sample
Pictures\\Winter.jpg"
));
    }
    currentPlayer = (currentPlayer%2)+1;
}

public static void main(String[] args) {
    ConnectFour game = new ConnectFour();
    ConnectFourGUI gui = new ConnectFourGUI();
    ConnectFourListener listener = new ConnectFourListener(game, gui);
}
}
```

set the  
value of a  
specified  
JLabel  
(either to  
a text  
value, or  
an image)

# ConnectFourListener.java

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

public class ConnectFourListener implements MouseListener {

    ConnectFourGUI gui;
    ConnectFour game;

    public ConnectFourListener(ConnectFour game, ConnectFourGUI gui) {
        this.game = game;
        this.gui = gui;
        gui.addListener(this);
    }

    public void mouseClicked(MouseEvent event) {
        JLabel label = (JLabel) event.getComponent();
        int column = gui.getColumn(label);
        int row = game.drop(column);
        if (row != -1) {
            gui.set(column, row);
        }
    }

    public void mousePressed(MouseEvent event) {
    }
    public void mouseReleased(MouseEvent event) {
    }
    public void mouseEntered(MouseEvent event) {
    }
    public void mouseExited(MouseEvent event) {
    }

}
```

store a reference to the GUI and game for later

what do you do when the mouse is clicked?

we're not concerned with these events, but since MouseListeners require them, leave them blank