CSCI-1302

Software Design
Spring 2012 – University of Georgia

Project 3

Four in a Row!

Goals

1) Implement a game using Java's Swing API and AWT
2) Implement event-handling in Java
3) Use the Java Stack collection in a typical UI manner

Points This project is worth 50 points

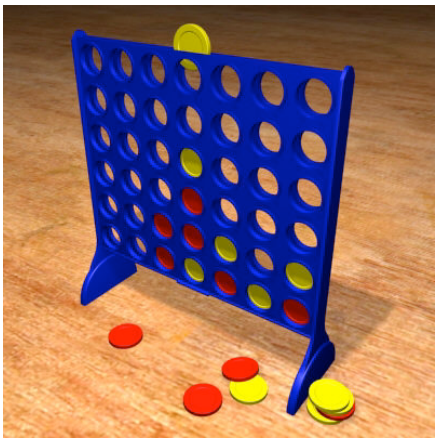
Due Date This project is due by 11 pm on March 9, 2012
The UML design document is due by 11 pm on Tuesday, Feb 28, 2012

Late Penalty Otherwise, 12% off the maximum original point value is deducted for each 24-hour period the assignment is late for up to two days late. Saturday/Sunday count as one day.

Collaboration Policy For this assignment, you may not discuss any code with anyone other than the instructor or teaching assistants.

The Java API, Java Tutorials, and the course textbooks are fair resources for this project.

Project Introduction



In this project, you'll get your feet wet with the graphical user interface (GUI) capabilities of Java. You will create a game using Java's AWT and Swing. Note, you may not use any GUI builder tools.

Four in a Row is a two-player game where each player takes turns dropping color-coded disks into a row of a grid. You may know this game as Milton Bradley's Connect Four. The object of the game is to first connect four of a player's disks next to each other vertically, horizontally, or diagonally.

In this project, you will create a graphical user interface to allow two humans to play this game.

Graphical User Interface Specifications

- The board shall of 7-column and 6-rows a piece.
- Generic pieces containing a blank space, red-player, and yellow-player are available on eLC; you may create your own pieces if you wish.
- The GUI must contain reset button to restart game play.
- The GUI must contain a leaderboard to keep score of each player. This is reset every time the application is quit and restarted, or the reset button is pressed.
- The GUI should contain a menu bar with a File and Help menu.

- File contains a Quit command
- Edit contains an Undo command that undoes the last move in reverse mode
- Help contains Get Started and About options
- The content for each of these menu items is fairly straightforward, as should the keyboard accelerators.

Game Play

- Players take turns using the mouse to click on the area above each column to indicate where they want to "drop" the disks.
- Once the player clicks, the disk drops down the column to the lowest unoccupied position
- If a player tries to click on a column already filled, you should display a dialog box indicating that this is an invalid move and to try an open column
- Once a move is made, your program must analyze whether a player has won. If so, change the color of the winning four-in-a-row to show the winning move and display a message congratulating the winner
- Some part of your UI needs to indicate whose turn it is (red or yellow)
- If at any point the reset button is pressed, the leader board is reset and the game board is reset.
- You can choose Undo to backtrack game play within a single game.

Input and Output Requirements

- Your driver class should be called `P3.java`
- You should also have a class for `GameBoard.java` and `FourInARow.java`
- You may construct any helper classes you wish using appropriate naming conventions
- Your program should also end/terminate only under the appropriate mechanism of someone choosing File -> Exit or using the keyboard accelerator
- There are no input arguments

Notes about GUI and Nike

If you are connecting remotely to Nike, you need to set up X11 forwarding using SSH in order to see the GUI program appear on your local machine. Here's a helpful link: http://tldp.org/HOWTO/XDMCP--HOWTO/ssh.html. If you are connecting via terminal on a Mac or Linux machine, simply use the –X flag when you are connecting to Nike. Alternately, you may opt to develop locally (i.e. using javac on your own Windows, Mac, or Linux computer).

Points

This project is worth 50 points towards your course grade. Grading of this programming project will use the following rubric:

Proper documentation (pre & post statements, commenting conditionals, not excessive commenting)	5 points
Reflection Documents Reflection of the initial design specifications after the code was implemented	5 points
UML Diagram	5 points
Passing Test Cases	20 points
Exceptional Condition Handling	15 points
Design and Usability	10 points

Total: 50 points

***Note**: This is our approximate grading distribution. Point values may vary.*

Submission Instructions

1. 1. Create a folder in an Nike account called **lastname_proj3** where lastname1and lastname2 is the last name of you.
   2. Copy all **thoroughly commented** Java source files in the folder created in step 1.
      3. Place a working makefile in the folder created in step 1 that has three directives:
         1. `compile:` compiles all of the source code
         2. `run:` runs an example of your program
         3. `clean:` removes all class files
      4. Add a `readme` file to the folder created in step 1 which has your name and clear instructions on how to compile and run your team's program.
      5. Remove all class files before submitting.
      6. Navigate to the parent directory of the folder created in step 1 on Nike, and issue the command below.

         ```
         submit  lastname_proj3  cs1302a
         ```

      7. If the submission was successful, then a file that begins with rec will be created in the submitted folder.