Algonquin College Logo

# SCHOOL OF ADVANCED TECHNOLOGY

### ICT - Applications & Programming

### Computer Engineering Technology – Computing Science



A21

Game MVC

Team:

[Student Name] - Id: [Student Id] / [Student Name] - Id: [Student Id]

Picross Proposal

***This template is suggested (not mandatory) to answer A21 Specification.***

|  |  |
| --- | --- |
| **Part**  **1** | **GUI Definition** |

*This template is very similar to your A11, but going deeper with the components and methods descriptin and dividing them into the MVC components.*

* 1. **MVC Details**

*Describe the way you can define the MVC components in your game.*

**Example** (from vision “top-down”)

Class: JFrame – Object: “GameFrame”

→ Class: JPanel → Object: “GameuBoard”

→ Class: JButtons → Objects: “BSave”, “BLoad”, etc.

→ Class: JLabel → Objects: “LabOperation”, “LabName”, etc.

…

Class: JFrame – Object: “picrossWindow”, “startWindow”, “instructionsWindow”

→ Class: JPanel → Object: “leftPanel”, “boardPanel”, “designMenuReturnPanel”, “languagePanel”, ”languageButtonPanel”, “gridSizeComboPanel”, “scorePanel”, ”timerPanel”, “buttonPanel”, “configurationPanel”, “titlePanel”, “historyPanel”, “controlPanel”, “colPanel”, “rowPanel”, “gridButtonPanel”, “splashTitlePanel”, “startPanel”, “configGrid”, “instructionsPanel”

→ Class: JButtons → Objects: “instructionsButton”, “instructionsBack”, “playButton”, “designButton”, “designBack”, “resetButton”,”solveButton”, “newBoardButton”, “[][] buttons”

→ Class: JLabel → Objects: “timerLabel”, “scoreLabel”, “gridSizeLabel”, “langLabel”, “colourLabel”, “picrossLabel”, “grid”, “titleLabel”, “instructionsLabel”

→ Class: JComboBox → Objects: “gridSizeCmbo”

→ Class: JRadioButton → Objects: “engRadio” , “frRadio”

→ Class: JCheckBox→ Objects: “markCheckBox”

→ Class: JTextArea → Objects: “history”

→ Class: JTextField → Objects: “scoreCounter”, “timerCounter”

→ Class: ButtonGroup → Objects: “langButtons”

* 1. **View Component**

*Describe how your interface should be organized using new components. Show the idea about your “top-down” organization.*

*\** ***TIP****: Review the components defined in the A11 and put them in the proper hierarchy (as you done in the A12).*

* + - ***(Play Frame)****:*

**Example** (from vision “top-down”)

Class: JFrame -> Object: PlayFrame (Play)

Class: JPanel -> Object: "leftPanel"

Class: JPanel -> Object: "scorePanel"

Class: JLabel -> Object: "scoreLabel"

Class: JTextField -> Object: "scoreCounter"

Class: JPanel -> Object: "timerPanel"

Class: JLabel -> Object: "timerLabel"

Class: JTextField -> Object: "timerCounter"

Class: JPanel -> Object: "buttonPanel"

Class: JButton -> Object: "resetButton"

Class: JButton -> Object: "solveButton"

Class: JButton -> Object: "newBoardButton"

Class: JButton -> Object: "instructionsButton"

Class: JPanel -> Object: "configurationPanel"

Class: JPanel -> Object: "languagePanel"

Class: JPanel -> Object: "languageButtonPanel"

Class: ButtonGroup -> Object: "langButtons"

Class: JRadioButton -> Object: "engRadio"

Class: JRadioButton -> Object: "frRadio"

* + - ***(titlePanel)****:*

**Example** (from vision “top-down”)

Class: JPanel -> Object: "titlePanel"

Class: ImageIcon -> Object: "picrossLogo"

Class: JLabel -> Object: "picrossLabel"→ Class: JPanel → Object: “splashPanel”

…

* + - ***(controlPanel)****:*

**Example** (from vision “top-down”)

Class: JPanel -> Object: "controlPanel"

Class: JPanel -> Object: "historyPanel"

Class: JTextArea -> Object: "history"

Class: JScrollePane -> Object: "scroll"

* + - ***(boardPanel)****:*

**Example** (from vision “top-down”)

Class: JPanel -> Object: "boardPanel"

Class: JPanel -> Object: "colPanel"

Class: JLabel -> Object: "grid"

Class: JPanel -> Object: "rowPanel"

Class: JLabel -> Object: "grid"

Class: JCheckBox -> Object: "Mark"

Class: JPanel -> Object: "gridButtonPanel"

Class: JButton newGridButton -> Object: "buttons"

* + - ***(menuBar)****:*

**Example** (from vision “top-down”)

Class: JMenuBar -> Object: "menuBar"

Class: JMenu -> Object: "gameMenu"

Class: JMenuItem -> Object: "newMenu"

Class: JMenuItem -> Object: "solutionMenu"

Class: JMenuItem -> Object: "exitMenu"

Class: JMenu -> Object: "helpMenu"

Class: JMenuItem -> Object: "colourMenu"

Class: JMenuItem -> Object: "aboutMenu"

* + - ***(instructionsWindow)****:*

**Example** (from vision “top-down”)

Class: JFrame -> Object: "instructionsWindow" (Instructions)

Class: JPanel -> Object: "instructionsPanel"

Class: JLabel -> Object: "instructionsLabel"

Class: JButton -> Object: "instructionsBack"

* + - ***(designFrame)****:*

**Example** (from vision “top-down”)

Class: JFrame -> Object: "designFrame" (Design)

Class: JPanel -> Object: "boardPanel"

Class: JPanel -> Object: "configGrid"

Class: JPanel -> Object: "languagePanel"

Class: JPanel -> Object: "gridSizeCmboPanel"

Class: JComboBox -> Object: options[] = {"5x5", "6x6", "7x7"}

Class: JLabel -> Object: "gridSize"

* + - ***(windowFrame)****:*

**Example** (from vision “top-down”)

Class: JFrame -> Object: "windowFrame" (Splash)

Class: JPanel -> Object: "splashTitlePanel"

Class: ImageIcon -> Object: "titleLogo"

Class: JLabel -> Object: "titleLabel"

Class: JFrame -> Object: "launcher" (Launcher)

Class: JPanel -> Object: "splashTitlePanel"

Class: ImageIcon -> Object: "titleLogo"

Class: JLabel -> Object: "titleLabel"

Class: JPanel -> Object: "startPanel"

Class: JButton -> Object: "designButton"

Class: JButton -> Object: "playButton"

* 1. **Controller Component**

*Describe aspects of your controller using, for example, one unique action command. Create the “map” to define functions with actions. This activity is to plan what will happen in your action. Ex:*

**Example** (from vision “top-down”)

Object: "instructionsButton"

-> Event: actionPerformed -> method: showInstructions()

**Example** (from vision “top-down”)

Object: "instructionsBack"

-> Event: actionPerformed -> method: Game()

**Example** (from vision “top-down”)

Object: "resetButton"

-> Event: actionPerformed -> method: resetBoard()

**Example** (from vision “top-down”)

Object: "solveButton"

-> Event: actionPerformed -> method: solveBoard()

**Example** (from vision “top-down”)

Object: "newBoardButton"

-> Event: actionPerformed -> method: makeNewBoard()

**Example** (from vision “top-down”)

Object: "designButton"

-> Event: actionPerformed -> method: makeDesignWindow()

**Example** (from vision “top-down”)

Object: "designBackButton"

-> Event: actionPerformed -> method: Game()

**Example** (from vision “top-down”)

Object: "playButton"

-> Event: actionPerformed -> method: makePlayWindow()

**Example** (from vision “top-down”)

Object: "engRadio"

-> Event: actionPerformed -> method: changeLangEng()

**Example** (from vision “top-down”)

Object: "frRadio"

-> Event: actionPerformed -> method: changeLangFr()

**Example** (from vision “top-down”)

Object: "markCheckBox"

-> Event: actionPerformed -> method: updateGameMode()

**Example** (from vision “top-down”)

Object: "gridSizeCmbo"

-> Event: actionPerformed -> method: updateBoardSize()

**Example** (from vision “top-down”)

Object: "buttons"

-> Event: actionPerformed -> method: updateData()

**Example**

Object: “aboutButton”

→ Event: actionPerformed → method: showAbout()

Etc.

// Future implementations - events (in controller)

**class** Controller **implements** ActionListener {

**public** **void** actionPerformed(ActionEvent e) {

Object eventSource = e.getSource();

// Button (about)

**if** (eventSource == *aboutButton*)

showAbout();

// Others…

|

|

* 1. **Model Component**

*Finally, what is your idea to define the model to be used in a “default” (randomized) game.*

**Example**

Data structure used:

→ Values: game → method: updateData()

→ Properties: dimension (int) → methods related: changeDimension(), getDimension(), …

→ Properties: Player (class)

→ Property: name (String) → methods related: changeDimension(), getDimension(),

|  |  |
| --- | --- |
| **Part**  **2** | **Implementation Design** |

* 1. **Game Evolution**
  + *Considering this new model, explain:*
    - *What are the differences between the original proposal (A11) and the current project to be developed (A21).*
    - *If so, explain why you need to do some adjustments.*
  1. **Others DP**
     + *Define (at least one) additional DP that you could use in your Game application.*
  + *Explain what is this DP and the reason why it could be recommended.*

**References**

*[Include eventual references used here]*

Algonquin College

Winter, 2022