Algonquin College Logo

# SCHOOL OF ADVANCED TECHNOLOGY

### ICT - Applications & Programming

### Computer Engineering Technology – Computing Science



A21

Game MVC

Team:

Isaac Proulx - Id: 041007853

Picross Proposal

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

* 1. **MVC Details**

(from vision “top-down”)

**Model** Class: Model – Object: model (POJO / Plain Java Old Object)

**View** Class: Game – Object: game (extends Application)

**Controller** Class: Controller – Object: controller (implements EventHandler<Event>) (responsible for all Actions)

* 1. **View Component**

(from vision “top-down”) (**WIP**)

Class: Scene – Object: “scene”

→ Class: GridPane → Object: “mainPane”

→ Class: MenuBar → Object: “menuBar” (newly added, meant to take the role of the options panel)

→ Class: Menu → Object: “fileMenu”

→ Class: Menu → Object: “languageMenu”

→ Class: MenuItem → Objects: “languageEnglish”, “languageFrench”

→ Class: MenuItem → Objects: “fileSave”, “fileLoad”

→ Class: Menu → Object: “gameMenu”

→ Class: Menu → Object: “dimensionMenu”

→ Class: MenuItem → Objects: (dimension options) (TBA)

→ Class: Menu → Object: “modeMenu”

→ Class: MenuItem → Objects: “modePlay” , “modeDesing”

→ Class: MenuItem → Objects: “gameNewGame”, “gameSolution”, “gameReset”

→ Class: VBox → Object: “optionsPanel”

→ Class: MenuButton → Object: “optionsMenu”

→ Class: MenuItem → Object: “language”

→ Class: MenuItem → Object: “newGame”

→ Class: HBox → Object: ”topPanel”

→ Class: VBox → Objects: ”colCount” ( there are multiple objects named “colCount”)

→ Class: StackPane → Objects: ”hintText” ( there are multiple objects named “hintText”)

→ Class: Text → Anonymous Object

→ Class: VBox → Object: ”leftPanel”

→ Class: HBox → Objects: ”rowCount” ( there are multiple objects named “rowCount”)

→ Class: StackPane → Objects: ”hintText” ( there are multiple objects named “hintText”)

→ Class: Text → Anonymous Object

→ Class: TilePane → Object: “gamePanel”

→ Class: StackPane → Objects: “tile” ( there are multiple objects named “tile”)

→ Class: VBox → Object: “controlPanel”

→ Class: ImageView → Object: “picrossLogo”

→ Class: HBox → Object: “timerPanel”

→ Class: Label → Object: “timerLabel”

→ Class: TextField → Object: “timerTime”

→ Class: HBox → Object: “scorePanel”

→ Class: Label → Object: “scoreLabel”

→ Class: TextField → Object: “scoreCount”

→ Class: TextArea → Object: “historyArea”

→ Class: HBox → Object: “buttonPanel”

→ Class: Button → Object: “resetButton”

→ Class: CheckBox → Object: “markBox”

* 1. **Controller Component**

**Controller**

Object: “tile”

→ Event: MouseEvent.MOUSE\_CLICKED → method: updateTile()

Object: “resetButton”

→ Event: ActionEvent.ACTION → method: resetGame()

Object: “markBox”

→ Event: ActionEvent.ACTION → method: toggleMark()

Object: “languageEnglish”

→ Event: ActionEvent.ACTION → method: setLanguage()

Object: “languageFrench”

→ Event: ActionEvent.ACTION → method: setLanguage()

Object: “fileSave”

→ Event: ActionEvent.ACTION → method: saveGame()

Object: “fileLoad”

→ Event: ActionEvent.ACTION → method: loadGame()

Object: “modePlay”

→ Event: ActionEvent.ACTION → method: setGameMode()

Object: “modeDesign”

→ Event: ActionEvent.ACTION → method: setGameMode()

Object: “gameNewGame”

→ Event: ActionEvent.ACTION → method: newGame()

Object: “gameSolution”

→ Event: ActionEvent.ACTION → method: showSolution()

Object: “gameReset”

→ Event: ActionEvent.ACTION → method: resetGame()

// Future implementations - events (in controller)

public class Controller implements *EventHandler*<Event> {

    private Game view;

    public Controller(Game *view*) {

*this*.*view* = view;

    }

    private void handleMenuItem(MenuItem *item*){

        String id = item.idProperty().get();

        switch(id){

            case "languageEnglish":{

                setLanguage("EN");

                break;

            }

            case "languageFrench":{

                setLanguage("FR");

                break;

            }

            case "fileSave":{

                saveGame();

                break;

            }

            case "fileLoad":{

                loadGame();

                break;

            }

            case "modePlay":{

                setGameMode(0);

                break;

            }

            case "modeDesign":{

                setGameMode(1);

                break;

            }

            case "gameNewGame":{

                newGame();

                break;

            }

            case "gameSolution":{

                showSolution();

                break;

            }

            case "gameReset":{

                resetGame();

                break;

            }

        }

    }

    private void handleTile(StackPane *tile*){

        int col = (int) tile.getProperties().get("col");

        int row = (int) tile.getProperties().get("row");

        updateTile(col,row);

    }

    private void handleButton(Button *button*){

        String id = button.idProperty().get();

        if(id=="resetButton"){

            resetGame();

            return;

        }

    }

    private void handleCheckBox(CheckBox *checkBox*){

        String id = checkBox.idProperty().get();

        if(id=="markBox"){

            toggleMark();

            return;

        }

    }

    @*Override*

    public void handle(Event *e*){

        Object eventSource = e.getSource();

        Class<? extends Object> sourceClass = eventSource.getClass();

        if(sourceClass == MenuItem.*class*) {

            handleMenuItem((MenuItem) eventSource);

            return;

        }

        if(sourceClass == StackPane.*class*){

*//the only clickable StackPanes should be tiles*

            handleTile((StackPane) eventSource);

            return;

        }

        if(sourceClass == CheckBox.*class*){

            handleCheckBox((CheckBox) eventSource);

            return;

        }

        if(sourceClass == Button.*class*){

            handleButton((Button) eventSource);

            return;

        }

    }

}

* 1. **Model Component**

Data structure used:

→ Property: dimension (int) → methods related: setDimension(), getDimension()

→ Property: board (char[][]) → methods related: updateTile(), getTile()

→ Property: points (int) → methods related: setPoints(), getPoints()

→ Property: time (int) → methods related: setTime(), getTime()

→ Property: language (int) → methods related: setLanguage(), getLanguage()

→ Properties: Player (class)

→ Property: name (String) → methods related: setName(), getName()

→ Property: maxScore (int) → methods related: setName(), getName()

→ Property: bestScoreTime (int) → methods related: setBestScoreTime (), getBestScoreTime ()

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

* 1. **Game Evolution**

Changes from original proposal (A11):

* The topPanel was incorrectly labeled as a VBox, it is now an HBox
* The leftPanel was incorrectly labeled as an HBox, it is now a VBox
* Added the menuBar as a replacement for the optionsPanel, should be easier to use
* Added the picrossLogo (in the controlPanel) which was missing before
* Minor layout adjustments to the topPanel and leftPanel
  + row/column hints are now in individual StackPanes so the numbers can be spaced easier
  1. **Others DP**

Algonquin College

Winter, 2022