<DROSSY>

Created by

Teetat Thamronglak 6632100121

Pasin Pornsiwakul 6632147721

2110215 Programming Methodology

Semester 2 Year 2023

Chulalongkorn University

<DROSSY>

Introduction

Drossy is a puzzle game. The objective of this game is to take the two drowsy characters (drossies) to bed. Each character will sleepwalk forward and your job is to direct them with placeable arrows to change their walking direction. Additionally, there are stars all across the levels, each of which is assigned to each character. If the drossies didn't collect all the stars, they would sleep into a nightmare and the level would be considered a fail. There are multiple levels, each with increasingly more difficult arrangements.

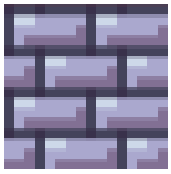
Game components

Humans

Beds

Each bed is occupied by each player. The players' goal is to reach their beds and sleep. If they reach another bed, they will wake up.

Wall

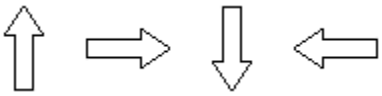


When the players hit the wall, they will wake up.

Stars

Each star is occupied by each player. The players' goal is to collect all of their stars. If they collect other stars, they will wake up.

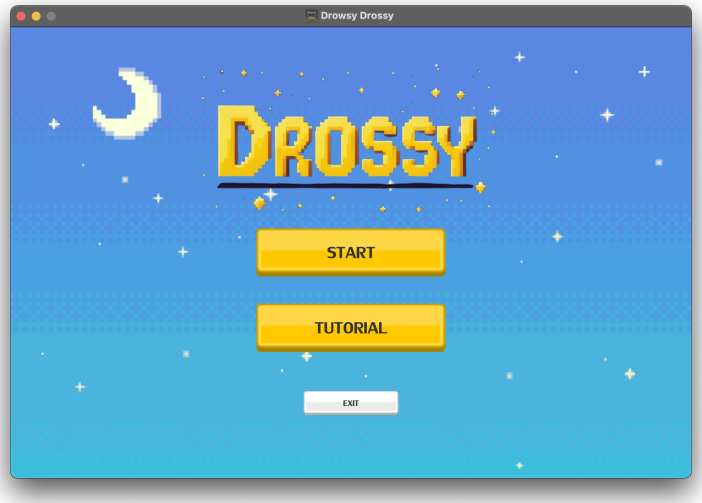
Arrows



When players land on the arrow, they change their direction.

How to play

<Main Menu>



After clicking the start button, the player will be sent to the level selection page. Aftering clicking tutorial button, the player will be sent to tutorial page

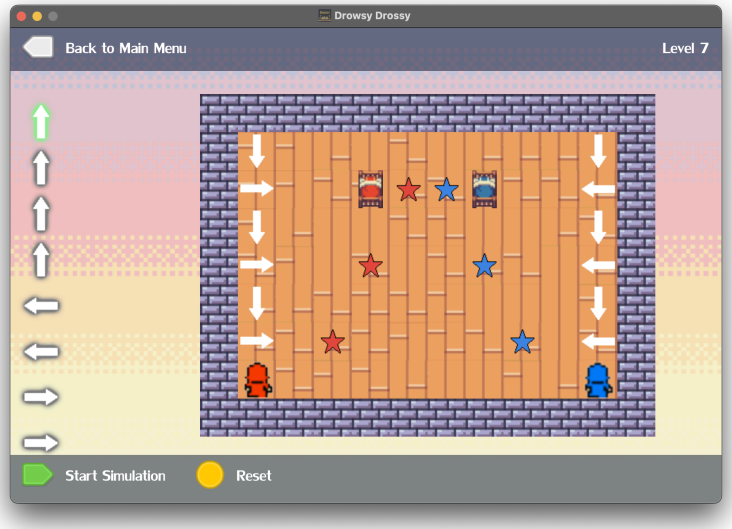
<Tutorial Page>

This page will provide information on how the game works and the backstory of the two characters.

<Level Selection Page>

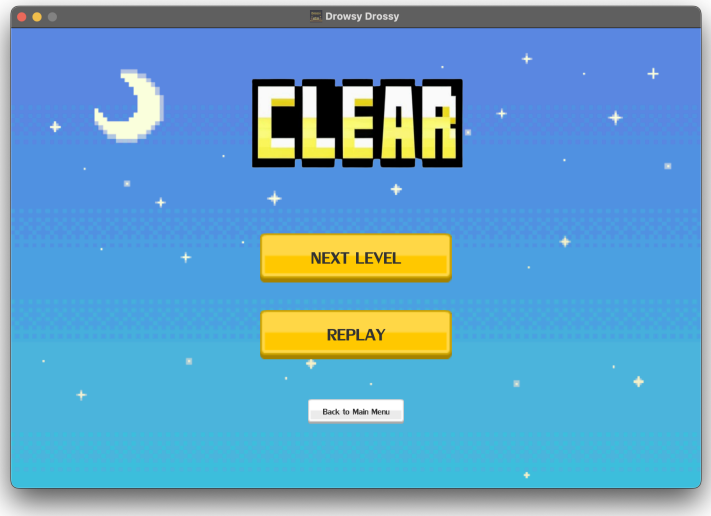
There will be a list of levels. The solved level will be shown as green, the attempted level will be shown as red, and the unattempted level will be shown as white. After clicking the level button, the player will proceed to the game page.

<Game Page>



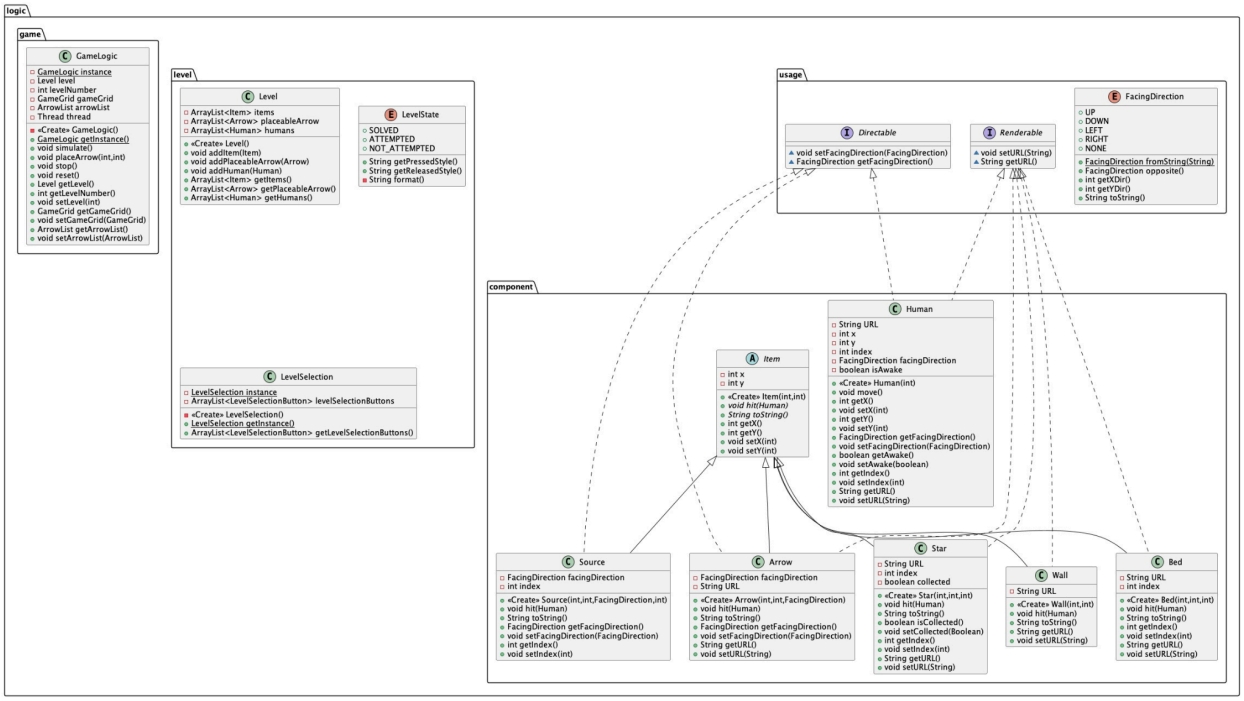
Inside a level will be objectives, the player must complete the objective of the game. After which they will go to the end game scene.

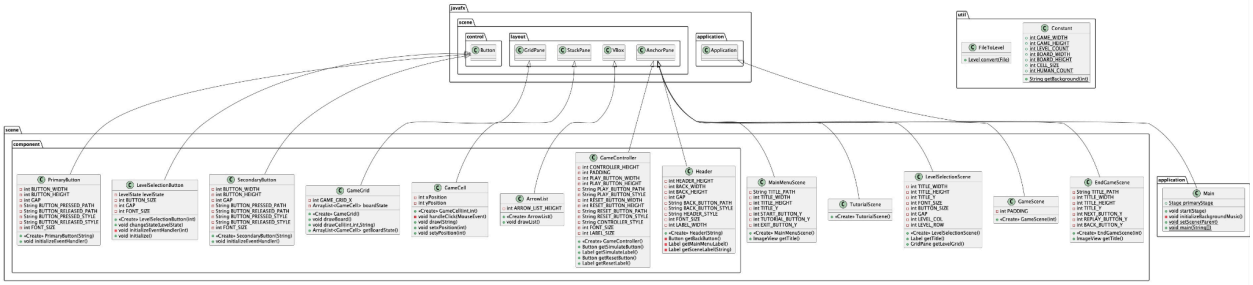
<End Game Page>



In this scene, player have completed an objective on the previous level and have choices to play the next level, replay or go back to main page

Class diagram





1. Package Application

This package contains the main class of this application.

1.1 Class Main

Fields

Name	Description
+ Stage primaryStage	The main stage of the application.

Methods

Method	Description
+ void start(Stage primaryStage)	Launch the application.
+ void initializeBackgroundMusic()	Play background music.
+ void setScene(Parent root)	Set the current scene to be the input root.
+ void main(String[] args)	-

2. Package logic

2.1 Package logic.usage

This package contains all interfaces and enums.

2.1.1 Enum **FacingDirection**

This enum contains 5 facing directions for the directable interface which are UP, DOWN, LEFT, RIGHT, and NONE.

Methods

Method	**Description**
+ FacingDirection fromString(String direction)	Return the facing direction from a string.
+ FacingDirection opposite()	Return the opposite direction of the current facing direction.
+ int getXDir()	Return the x direction according to the facing direction.
+ int getYDir()	Return the y direction according to the facing direction.
+ String toString()	Return the string representation of the facing direction.

2.1.2 Interface **Directable**

This interface is for the directed item.

Methods

Method	**Description**
void setFacingDirection(FacingDirection facingDirection)	Set the object's facing direction.
FacingDirection getFacingDirection()	Return the object's facing direction.

2.1.3 Interface **Renderable**

This interface is for the item that can be rendered.

Methods

Method	Description
void setURL(StringURL)	Set the object's Image URL.
String getURL()	Return the object's Image URL.

2.2 Package logic.component

This package contains all components in the game

2.2.1 Class **Human** implements **Directable, Renderable**

Fields

Name	Description
- String URL	Image URL for rendering.
- int x	x-position.
- int y	y-position.
- int index	Human's index.
- FacingDirection facingDirection	Human's facing direction.
- boolean isAwake	Store whether the human is awake.

Methods

Method	Description
+ Human(int index)	Initialize the human with the index. Set the image URL based on the index.
+ void move()	Move the human according to the facing direction.
+ getter/setter	getter/setter.

*2.2.2 Abstract Class **Item***

This class is a generalized class of all items in the game.

Fields

Name	**Description**
- int x	x-position.
- int y	y-position.

Methods

Method	**Description**
+ Item(int x, int y)	Constructor.
+ abstract void hit(Human human)	Handle the hit event when the item is hit by a human.
+ abstract String toString()	Return the string representation of the item.
+ getter/setter	getter/setter.

2.2.3 Class **Source** extends **Item** implements **Directable**

Fields

Name	**Description**
- int index	Source's index.
- FacingDirection facingDirection	Source's facing direction.

Methods

Method	**Description**
+ Source(int x, int y, FacingDirection facingDirection, int index)	Constructor.
+ void hit(Human human)	Do nothing.
+ String toString()	Return "Source".

+ getter/setter	getter/setter.

2.2.4 Class **Bed** extends **Item** implements **Renderable**

Fields

Name	**Description**
- String URL	Image URL for rendering.
- int index	Bed's index.

Methods

Method	**Description**
+ Bed(int x, int y, int index)	Constructor. Set the image URL based on the index.
+ void hit(Human human)	Stop the human from moving. If the human is not the same as the bed, wake the human up.
+ String toString()	Return "Bed".
+ getter/setter	getter/setter.

2.2.5 Class **Wall** extends **Item** implements **Renderable**

Fields

Name	**Description**
- String URL	Image URL for rendering.

Methods

Method	**Description**
+ Wall(int x, int y)	Constructor.

+ void hit(Human human)	Stop the human from moving, and wake the human up.
+ String toString()	Return “Wall”.
+ getter/setter	getter/setter.

2.2.6 Class **Star** extends **Item** implements **Renderable**

Fields

Name	**Description**
- String URL	Image URL for rendering.
- int index	Star’s index.
- boolean collected	Store whether the star is collected.

Methods

Method	**Description**
+ Star(int x, int y, int index)	Constructor. Set the image URL based on the index.
+ void hit(Human human)	If the human is not the same as the star, wake the human up. Otherwise, set the star as collected.
+ String toString()	Return “Star”.
+ getter/setter	getter/setter.

2.2.7 Class **Arrow** extends **Item** implements **Directable, Renderable**

Fields

Name	**Description**
- String URL	Image URL for rendering.
- FacingDirection facingDirection	Arrow’s facing direction.

Methods

Method	Description
+ Arrow(int x, int y, FacingDirection facingDirection)	Constructor. Set the image URL based on the facing direction.
+ void hit(Human human)	Change the facing direction of the human to the facing direction of the arrow.
+ String toString()	Return “Arrow”.
+ getter/setter	getter/setter.

2.3 Package logic.level

This package contains a level class.

2.3.1 Class **Level**

Fields

Name	Description
- ArrayList<Item> items	The array list contains all items on the board.
- ArrayList<Arrow> placeableArrows	The array list contains all placable arrows.
- ArrayList<Human> humans	The array list contains all humans.

Methods

Method	Description
+ Level()	Constructor.
+ void addItem(Item item)	Add item to items.
+ void addArrow(Arrow arrow)	Add arrow to arrows.
+ void addHuman(Human human)	Add human to humans.

+ getter	getter.

2.3.2 Class **LevelSelection**

Fields

Name	Description
- LevelSelection instance	Level selection instance.
- ArrayList<LevelSelectionButton> levelSelectionButtons	Array list of level selection buttons.

Methods

Method	Description
- LevelSelection()	Constructor.
+ LevelSelection getInstance()	Get the instance of the LevelSelection. If the instance is null, create a new instance.
+ ArrayList<LevelSelectionButton> getLevelSelectionButtons()	Getter for levelSelectionButtons.

2.3.3 Enum **LevelState**

This enum contains 3 level states for the level selection buttons which are NOT_ATTEMPTED, ATTEMPTED, and SOLVED.

Methods

Method	Description
+ String getPressedStyle()	Returns style of level selection button when pressed for each state.
+ String getReleasedStyle()	Returns style of level selection button when released for each state.
- String format()	Formats the enumerator into String.

2.4 Package logic.game

This package contains all game logic.

2.4.1 Class **GameLogic**

Fields

Name	Description
- GameLogic instance	Game logic instance.
- Level level	Current level.
- int levelNumber	Current level number.
- GameGrid gameGrid	Current level's game grid.
- ArrowList arrowList	Current level's arrow list.
- Thread thread	The thread that is currently used for simulating.

Methods

Method	Description
- GameLogic()	Constructor.
+ GameLogic getInstance()	Get the instance of the GameLogic. If the instance is null, create a new instance.
+ void placeArrow(int x, int y)	If there is no item at the given position, place the first arrow in arrow list at the given position and remove it from the arrow list.
+ void simulate()	Simulate the game using a thread.
+ void stop()	If the thread is running, interrupt the thread.
+ void reset()	Reset the level.
+ getter/setter	getter/setter.

3. Package scene

3.1 Package scene.component

This package contains the components that are used in the scenes.

3.1.1 Class **ArrowList** extends **VBox**

This class represents the arrow list in game component scenes.

Fields

Name	Description
- final String ARROW_LIST_HEIGHT	Height of arrow list in pixels.

Methods

Method	Description
+ ArrowList(int level)	Constructor: creates a new arrow list component alongside its styling.
+ void drawList()	Get the current placeable arrow list from gameLogic and draw them.

3.1.2 Class **GameCell** extends **StackPane**

This class represents the game cell component in game scenes.

Fields

Name	Description
- int xPosition	The x coordinate of the cell.
- int yPosition	The y coordinate of the cell.

Methods

Method	Description
+GameCell(int x, int y)	Constructor: creates a new game cell component alongside its styling.
- handleClick()	Handle click action: place an arrow on the game cell if there are any left in GameLogic's placeable arrow.
+ draw(String path)	Load an image from the path onto the game cell.
+ setter	setter for xPosition and yPosition

3.1.3 Class **GameController** extends **AnchorPane**

This class represents the game controller component in game scenes.

Fields

Name	Description
- final int CONTROLLER_HEIGHT	Controller height.
- final int PADDING	Padding.
- final int PLAY_BUTTON_WIDTH	Play button width.
- final int PLAY_BUTTON_HEIGHT	Play button height.
- final String PLAY_BUTTON_PATH	Resource path for play button.
- final String PLAY_BUTTON_STYLE	CSS styling for play button.
- final int RESET_BUTTON_WIDTH	Reset button width.
- final int RESET_BUTTON_HEIGHT	Reset button height.
- final String RESET_BUTTON_PATH	Resource path for play button.
- final String RESET_BUTTON_STYLE	CSS styling for reset button.
- final String CONTROLLER_STYLE	CSS styling for game controllers.
- final int LABEL_SIZE	The label's width in pixels.

Name	Description
- final int FONT_SIZE	The font size of the label's text.

Methods

Method	Description
+ GameController()	Constructor: creates a new game controller component alongside its styling.
+ Button getSilumateButton()	Get a simulate button as JavaFX Button.
+ Label getSimulateLabel()	Get a simulate label as JavaFX Label.
+ Button getResetButton()	Get a reset button as JavaFX Button.
+ Label getResetLabel()	Get a reset label as JavaFX Label.

3.1.4 Class **GameGrid** extends **GridPane**

This class represents the game grid component in game scenes.

Fields

Name	Description
- final String GAME_GRID_X	Game grid's X position in pixels from the left.
- ArrayList<GameCell> boardState	Board state of the game containing each game cell.

Methods

Method	Description
+GameGrid()	Constructor: creates a new game grid component alongside its styling.
+ void drawBoard()	Draw the current board from GameLogic. This method will utilize drawCell.

Method	Description
+ void drawCell(int x, int y, String path)	Draw the cell at position x, y with image from path.
+ ArrayList<GameCell> getBoardState()	Getter for boardState.

3.1.5 Class **Header** extends **AnchorPane**

This class represents the header component in various scenes.

Fields

Name	Description
- final int HEADER_HEIGHT	Header's height in pixels.
- final int BACK_WIDTH	Back button's width in pixels.
- final int BACK_HEIGHT	Back button's height in pixels.
- final int GAP	Gap between each component in pixels.
- final String BACK_BUTTON_PATH	Path to back button's image.
- final String BACK_BUTTON_STYLE	CSS styling of back button. This field utilizes BACK_BUTTON_PATH.
-final String HEADER_STYLE	CSS styling of header.
- final int FONT_SIZE	Font size each label.
- final int LABEL_WIDTH	The width of each label in pixels.

Methods

Method	Description
+ Header(String text)	Constructor: creates a new header component alongside its styling. The input text will be displayed on the right-side of the header.
- Button getBackButton()	Get back button as JavaFX Button component.

Method	Description
- Label getMainMenuLabel()	Get main menu label as JavaFX Label component with text “Back to Main Menu”.
- Label getSceneLabel(String text)	Get scene label as JavaX Label component. The input text will be displayed on the label.

3.1.6 Class **LevelSelectionButton** extends **Button**

This class represents the level selection button component in the level selection scene.

Fields

Name	Description
- LevelState levelState	The level state of the current button.
- final int BUTTON_SIZE	The width and height of the level selection button in pixels.
- final int GAP	The difference between the height of the button when pressed and released. This field will be utilized in initializeEventHandler.
- final int FONT_SIZE	The font size of the button’s text.

Methods

Method	Description
+ LevelSelectionButton(int level)	Constructor: creates a new level selection button. The input level will determine which level this button takes to.
+ void changeState(LevelState levelState)	Changes the state of a level from not-attempted -> attempted -> solved. The states cannot be changed backwards.
- void initializeEventHandler(int level)	Initialized a shared event handler of level selection button: drop shadow on mouse entered, null on mouse exited, pressed and released styles, and showing GameScene to the input’s level when clicked.

Method	Description
- void initialize()	Initializes the level selection button styling.

3.1.7 Class **PrimaryButton** extends **Button**

This class represents the primary button component in various scenes.

Fields

Name	Description
- final int BUTTON_WIDTH	The width of the primary button in pixels.
- final int BUTTON_HEIGHT	The height of the primary button in pixels.
- final int GAP	The difference between the height of the button when pressed and released. This field will be utilized in initializeEventHandler.
- final String BUTTON_PRESSED_PATH	Path to primary button's pressed image.
- final String BUTTON_RELEASED_PATH	Path to primary button's released image.
- final String BUTTON_PRESSED_STYLE	CSS styling of the primary button's pressed style. This field utilizes BUTTON_PRESSED_PATH.
- final String BUTTON_RELEASED_STYLE	CSS styling of the primary button's released style. This field utilizes BUTTON_RELEASED_PATH.
- final int FONT_SIZE	The font size of the button's text.

Methods

Method	Description
- void initializeEventHandler(int level)	Initialized a shared event handler of level selection button: drop shadow on mouse entered, null on mouse exited, and pressed and released styles.

3.1.8 Class **SecondaryButton** extends **Button**

This class represents the secondary button component in various scenes.

Fields

Name	Description
- final int BUTTON_WIDTH	The width of the secondary button in pixels.
- final int BUTTON_HEIGHT	The height of the secondary button in pixels.
- final int GAP	The difference between the height of the button when pressed and released. This field will be utilized in initializeEventHandler.
- final String BUTTON_PRESSED_PATH	Path to secondary button's pressed image.
- final String BUTTON_RELEASED_PATH	Path to secondary button's released image.
- final String BUTTON_PRESSED_STYLE	CSS styling of the secondary button's pressed style. This field utilizes BUTTON_PRESSED_PATH.
- final String BUTTON_RELEASED_STYLE	CSS styling of the secondary button's released style. This field utilizes BUTTON_RELEASED_PATH.
- final int FONT_SIZE	The font size of the button's text.

Methods

Method	Description
- void initializeEventHandler(int level)	Initialized a shared event handler of level selection button: drop shadow on mouse entered, null on mouse exited, and pressed and released styles.

3.2 Class **EndGameScene** extends **AnchorPane**

This class represents the end game scene.

Fields

Name	Description
- final String TITLE_PATH	Path to title image.
- final int TITLE_WIDTH	Title's width in pixels.
- final int TITLE_HEIGHT	Title's height in pixels.
- final int TITLE_Y	Title's Y position in pixels from the top.
- final int NEXT_BUTTON_Y	Start button's Y position in pixels from the top.
- final int REPLAY_BUTTON_Y	Replay button's Y position in pixels from the top.
- final int BACK_BUTTON_Y	Tutorial button's Y position in pixels from the top.

Methods

Method	Description
+ EndGameScene(int level)	Constructor: creates a new end game scene alongside its components.
+ ImageView getTitle()	Get clear title as JavaFX ImageView component.

3.3 Class **GameScene** extends **AnchorPane**

This class represents the game scene.

Fields

Name	Description
- final int PADDING	Padding from the left of the placeable arrow list.

Methods

Method	Description
+ GameScene(int level)	Constructor: creates a new game scene alongside its components. Level that loads will be the input level.

3.4 Class **LevelSelectionScene** extends **AnchorPane**

This class represents the level selection scene.

Fields

Name	Description
- final int TITLE_WIDTH	Title's width in pixels.
- final int TITLE_HEIGHT	Title's height in pixels.
- final int TITLE_Y	Title's Y position in pixels from the top.
- final int FONT_SIZE	Title's font size.
- final int BUTTON_SIZE	Level selection button's width and height in pixels.
- final int GAP	VGap and HGap for level grid in pixels.
- final int LEVEL_ROW	Number of rows in level grid.
- final int LEVEL_COL	Number of columns in level grid.

Methods

Method	Description
+ LevelSelectionScene()	Constructor: creates a new level selection scene alongside its components.
+ Label getTitle()	Get game title as JavaFX Label component with text "Pick a level".
+ GridPane getLevelGrid()	Get level grid as JavaFX GridPane component with level selection button component in each grid slot.

3.5 Class **MainMenuScene** extends **AnchorPane**

This class represents the main menu scene.

Fields

Name	Description
- final String TITLE_PATH	Path to title image.
- final int TITLE_WIDTH	Title's width in pixels.
- final int TITLE_HEIGHT	Title's height in pixels.
- final int TITLE_Y	Title's Y position in pixels from the top.
- final int START_BUTTON_Y	Start button's Y position in pixels from the top.
- final int TUTORIAL_BUTTON_Y	Tutorial button's Y position in pixels from the top.
- final int EXIT_BUTTON_Y	Exit button's Y position in pixels from the top.

Methods

Method	Description
+ MainMenuScene()	Constructor: creates a new main menu scene alongside its components.
+ ImageView getTitle()	Get game title as JavaFX ImageView component.

3.6 Class **TutorialScene** extends **AnchorPane**

This class represents the tutorial scene.

Method	Description
+ TutorialScene()	Constructor: creates a new tutorial scene alongside its components.

4. Package util

4.1 Class **Constant**

This class represents shared constants used throughout the application

Fields

Name	Description
+ final int GAME_WIDTH	Game width in pixels.
+ final int GAME_HEIGHT	Game height in pixels.
+ final int LEVEL_COUNT	Number of levels in the game.
+ final int BOARD_WIDTH	The width of the game board in cells.
+ final int BOARD_HEIGHT	The height of the game board in cells.
+ final int CELL_SIZE	Cell width and height in pixels.
+ final int HUMAN_COUNT	Number of humans in each level.

Methods

Method	Description
+ String getBackground(int index)	Get a path to the background image index.

4.2 Class **FileToLevel**

This Class provides a File to Level conversion utility.

Methods

Method	Description
+ Level convert(InputStream inputStream)	Convert level info from .txt file to Level class. This function takes an input file that will be converted, the format of this file is as follows : first line: N -> number of items inside the level. the following N lines: x y type [human index] [facing

Method	Description
	direction] -> the x and y position of an object and type of object. Some objects may require [human index] or [facing direction]. next line: M -> number of placeable arrows. the following M lines: facing direction -> the facing direction of each arrow. And will return a level object with the following properties