# IsoSlime

Documentation

Created by

Sorawit Kamphoi 6432179621

2110215 Programming Methodology

Semester 2 Year 2021

Chulalongkorn University

# IsoSlime

## 1. Introduction

IsoSlime is a puzzle and short story game inspired from Sokoban (Famous box sliding puzzle game), ice sliding puzzle, and more puzzles games. The main objective of this game is completing puzzles and challenging player to beat as many puzzles as you can. This game will be shown with isometric perspective.

## 2. Game Explanation

### 2.1. Overview of Game

The game is presented in isometric perspective. The goal of the game is to go to the finish gate by passing puzzles. The maximum size of puzzle map is 10 x 10 dimension.

### 2.2. Modes

There are 2 modes for the game

1) Story Mode — Player can play with short stories for 50 scenes.
2) Puzzle Mode — Beat as many levels as possible with time limitation.

In puzzle mode, the time limit is 2 minutes and player will receive 30 seconds per 1 level completed.

## 2.3. Main Character

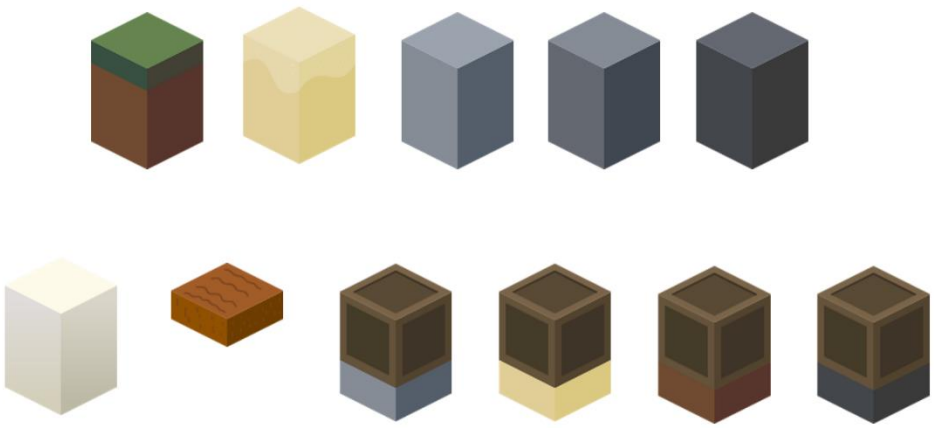
The main character of this game is pink slime. Player can control character by press mouse (which will be told later).

## 2.4. Tiles

The tiles are platform that player can walk, slide, interact or prevent from moving. There are 4 kinds of tiles

### 2.4.1. Normal Tiles

These tiles don't have any ability, but player can walk through. These are all types of normal tiles.

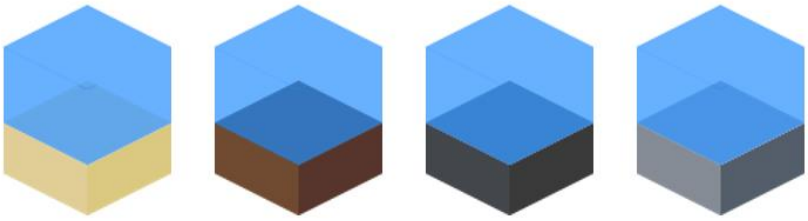


### 2.4.2. Ice

The ice allows player and crate to slide. (The detail of sliding will be told later)

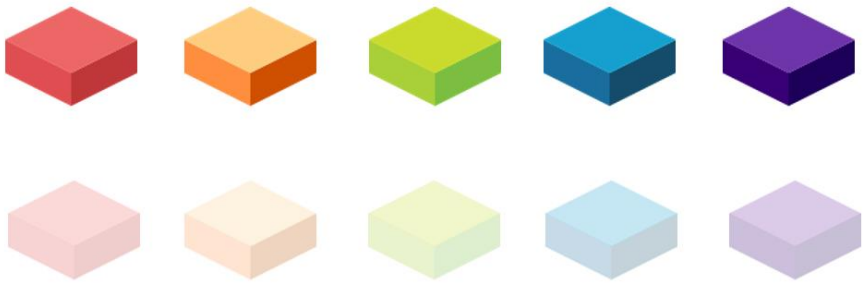
### 2.4.3. Water Tiles

These tiles don't let player pass through but allow crate to dropped and player can move after it dropped.



### 2.4.4. Color Platform

These tiles can be switched between active and inactive. Player can move through active platform but can't on inactive platform.



## 2.5. Wooden Crate

Player can push the crate, also these crated can be dropped in water. (Which will be told later)

## 2.6. Level Gate

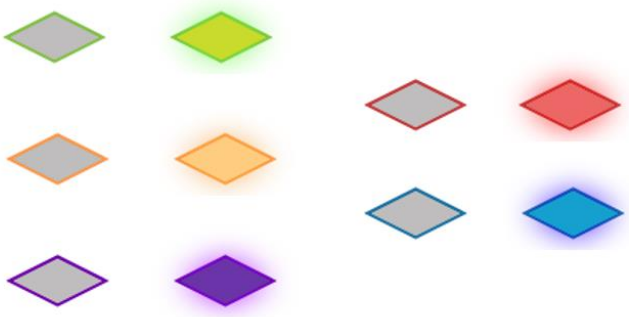
This gate act like finish line of the level, player must go to this gate to complete that level.

## 2.7. Tree

These trees act like obstacles but player can chop them down.

## 2.8. Button Sensor

Player or crate can active these sensors. The platform with same color as sensor will change their status.



## 2.9. Items

There are 3 kinds of items in this game which can be selected in inventory pane.



- Magic Wand

Player can teleport yourself within range of 2 blocks from player.

- Axe

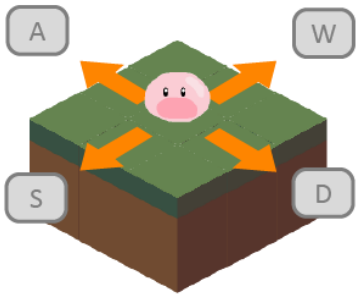
Player can chop the tree by using axe, after chop tree, 1 wood will be received.

- Wood

Player can place wood on empty space in the map to create wooden platform.

## 2.10. How to Play

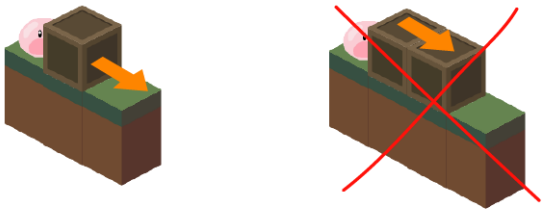
Player can press W A S or D to move character. The goal of this game is reaching level gate.




Press W A S D to move

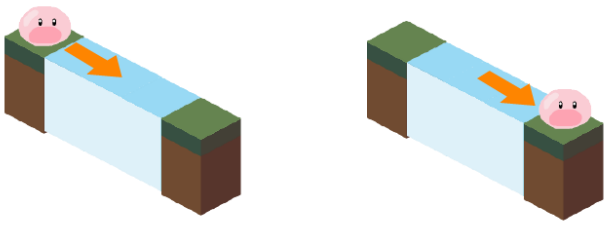
Your goal is to reach this gate

Player can push crate but player cannot push 2 or more crates at once



Crate can be dropped on water, after crate dropped, player can pass through the crate.

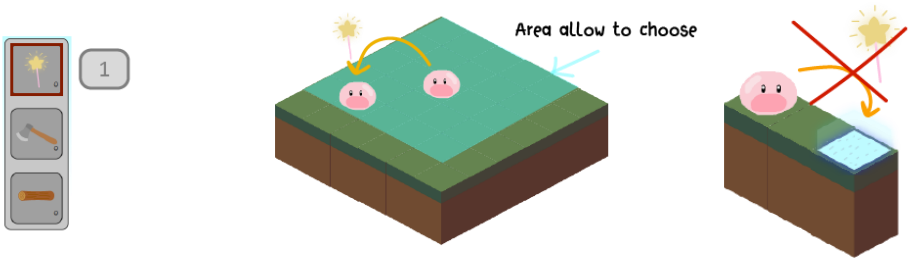
Player can slide on the ice, player will continually slide until land on normal tile or fall from map.



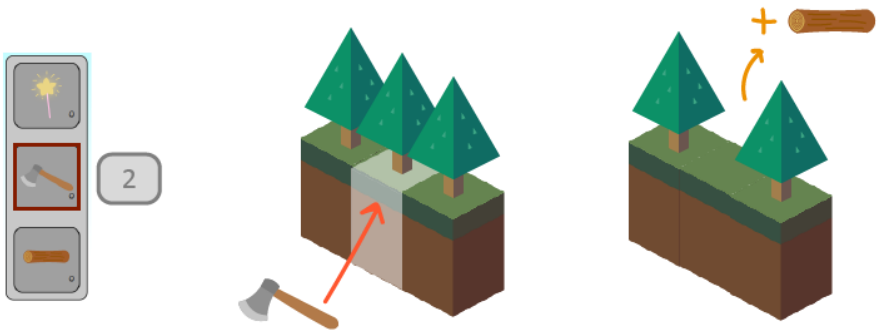
Player can obtain items from floor. Player can use items by press 1 / 2 / 3 or click on the inventory. There are 3 types of items available.


First item is “Magic Wand”, player can teleport yourself in the range of 2 blocks from yourself. Select magic wand and click on tile to teleport.

Warning: Player cannot teleport to level gate directly!

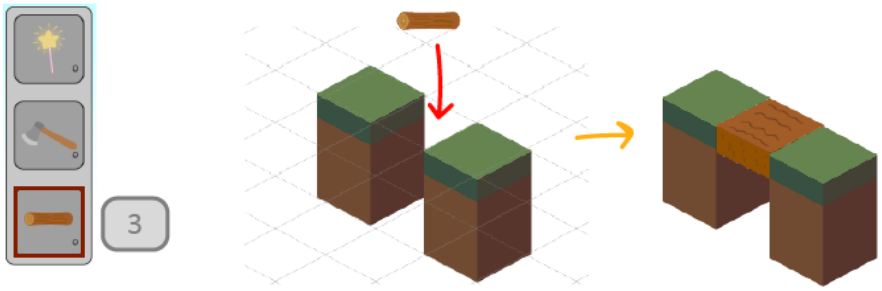



Second item is “Axe”, player can chop tree down and player will obtain 1 wood. Select the axe and click on tree that you want to chop.

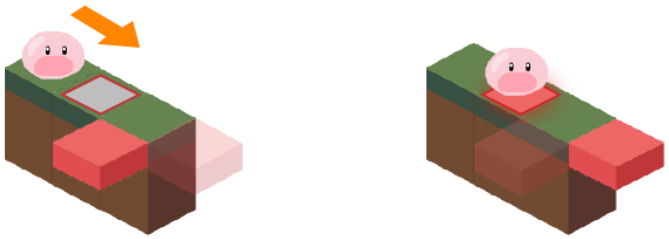


Last item is “Wood”, player can place wood on empty tiles to build wooden platform. Select wood and click on empty tile to build platform.

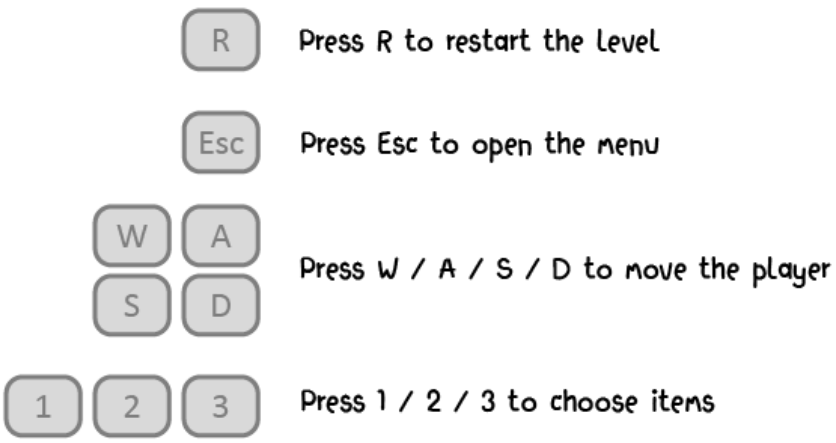
Warning: You cannot place wood on water!




The button sensor allow player to switch status of color platform. The sensor will also work if the crate press on sensor.



## 2.11. Keyboard control



# 3. Screen of Game

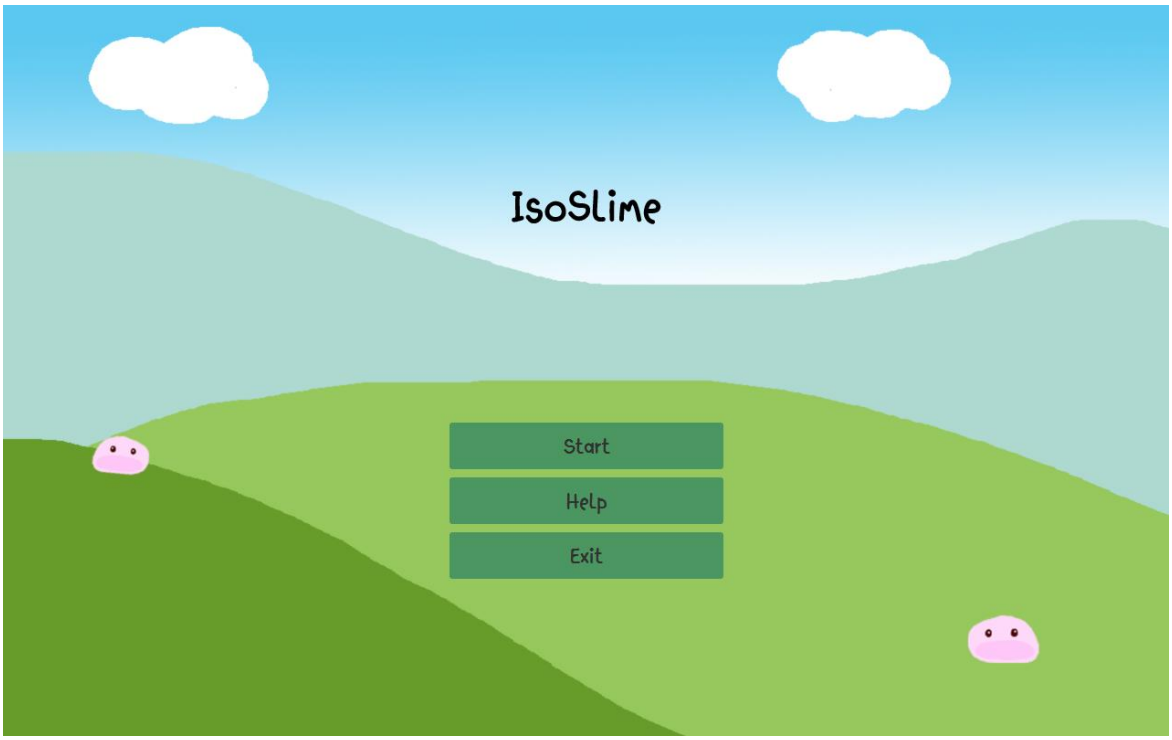
## 3.1. Main Menu

This screen consists of start, help, and exit buttons. This is also first screen for everyone.

Start button: Direct to mode selecting screen

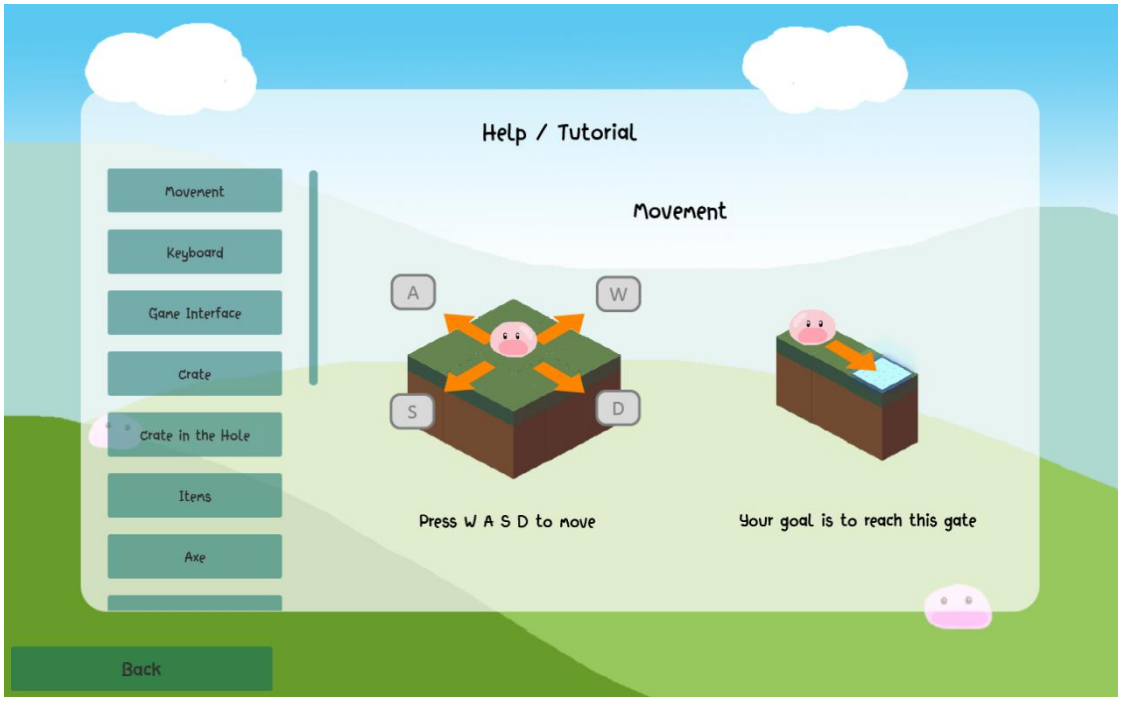
Help button: Direct to help and tutorial screen

Exit button: Exit from game



## 3.2. Help and Tutorial

This screen will tell everything you need to know for playing this game. You can select the thing you want to know at left side and the tutorial will be display at right side. You can press back at bottom left to return to main menu.

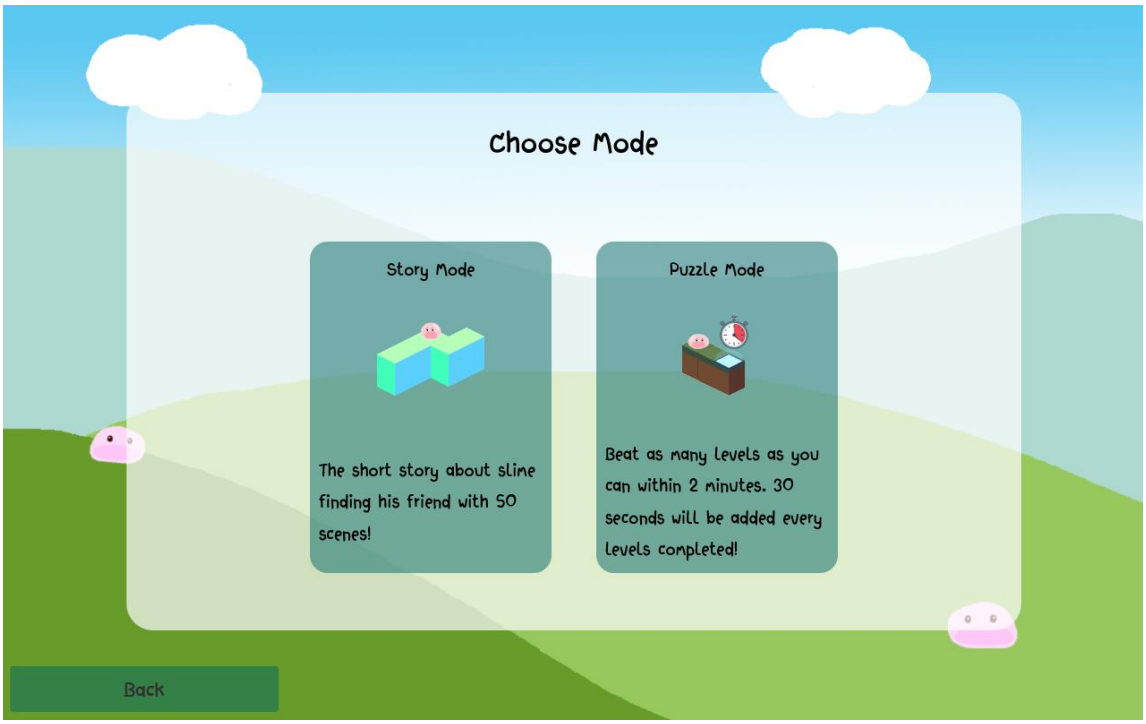


### 3.3. Mode Selecting

After player press start button, player can choose mode for gameplay.

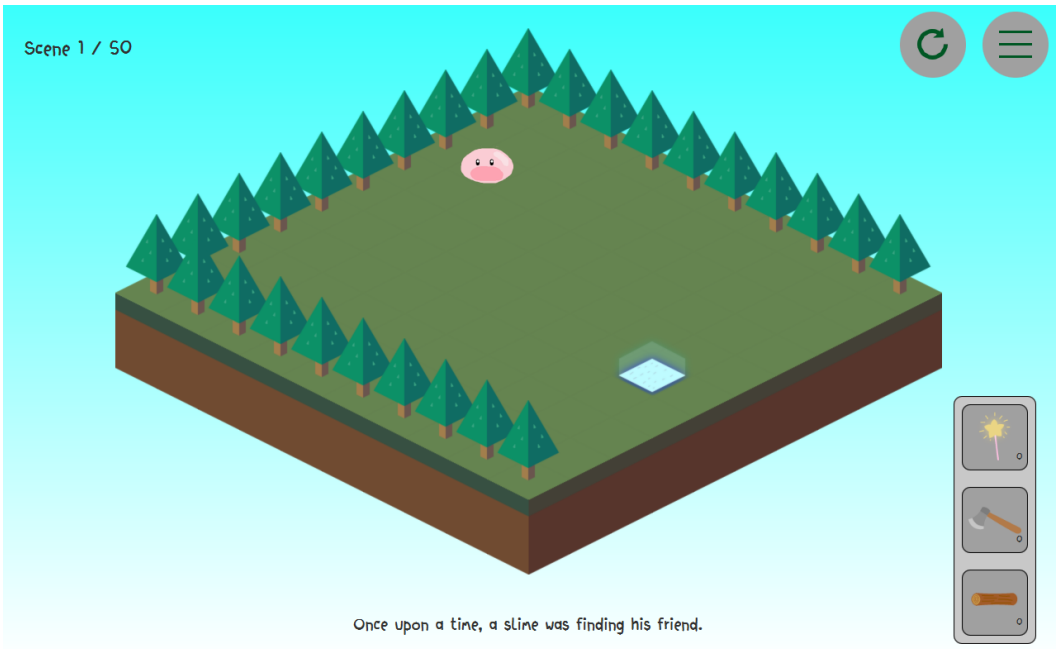
**Story Mode** : Player will play as slime who is lost in forest and tried to finding his friend. There are 50 scenes to play.

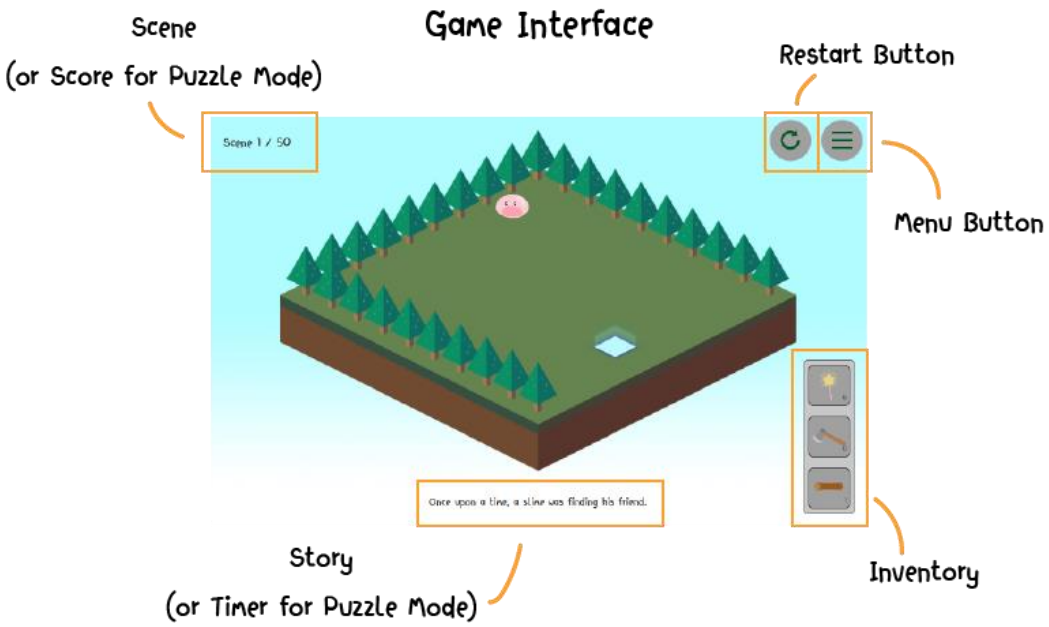
**Puzzle Mode** : Beat as many level as possible within 2 minutes. Every level completed, player will receive 30 seconds. All level is random!



### 3.4. Game Screen

The center of the screen is puzzle that you must solve. The bottom right is inventory pane which player can click on items to use. The bottom center is story text which tell story of the game in story mode or tell time left for puzzle mode. The top left is scene text which tell scene number for story mode or score for puzzle mode. In the top right, you can click on menu bar.

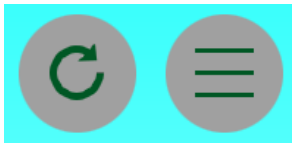


On the menu bar, there are 2 buttons which are restart and menu button.

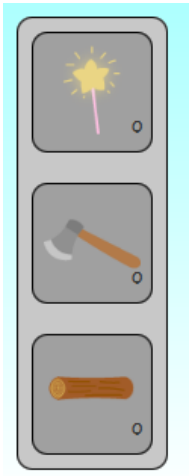
Restart button: The game will restart.

Menu button: Showing menu panel.



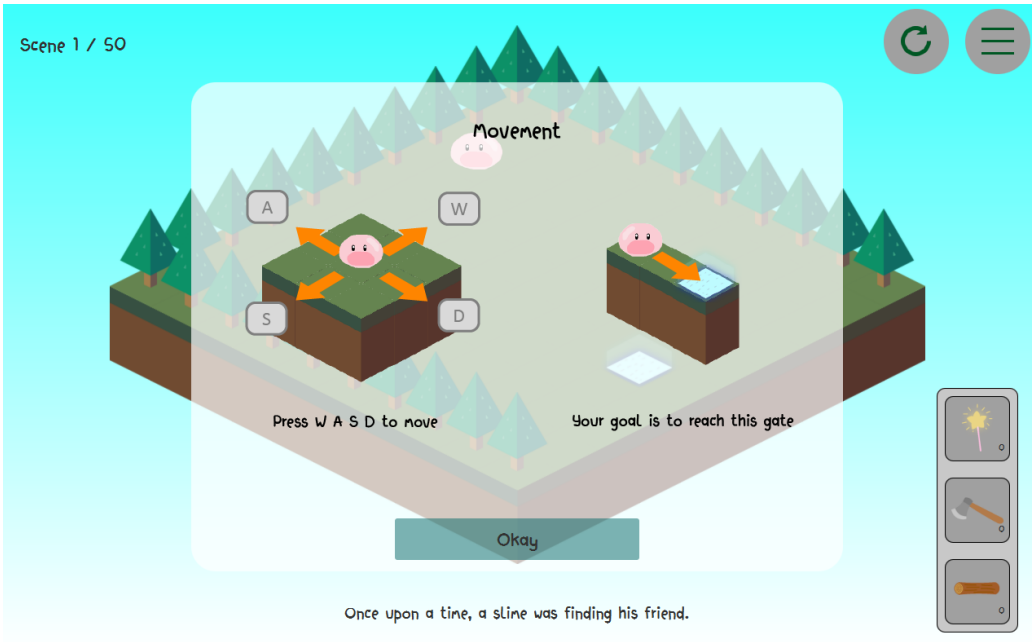
In menu panel, there are 3 buttons available, resume, restart, exit button.


The inventory bar is at bottom right of screen, player can click on items and use. The number at bottom right of each items tell amount of items left.



### 3.5. Tutorial in Story Mode

While playing story mode, some tutorial will be shown to tell information player must know for completing that level. Click Okay button to continue.




### 3.6. Winning for Story Mode

After player completed level 50, the congratulation screen will be shown. There are 2 buttons available, restart the story or exit the story mode.

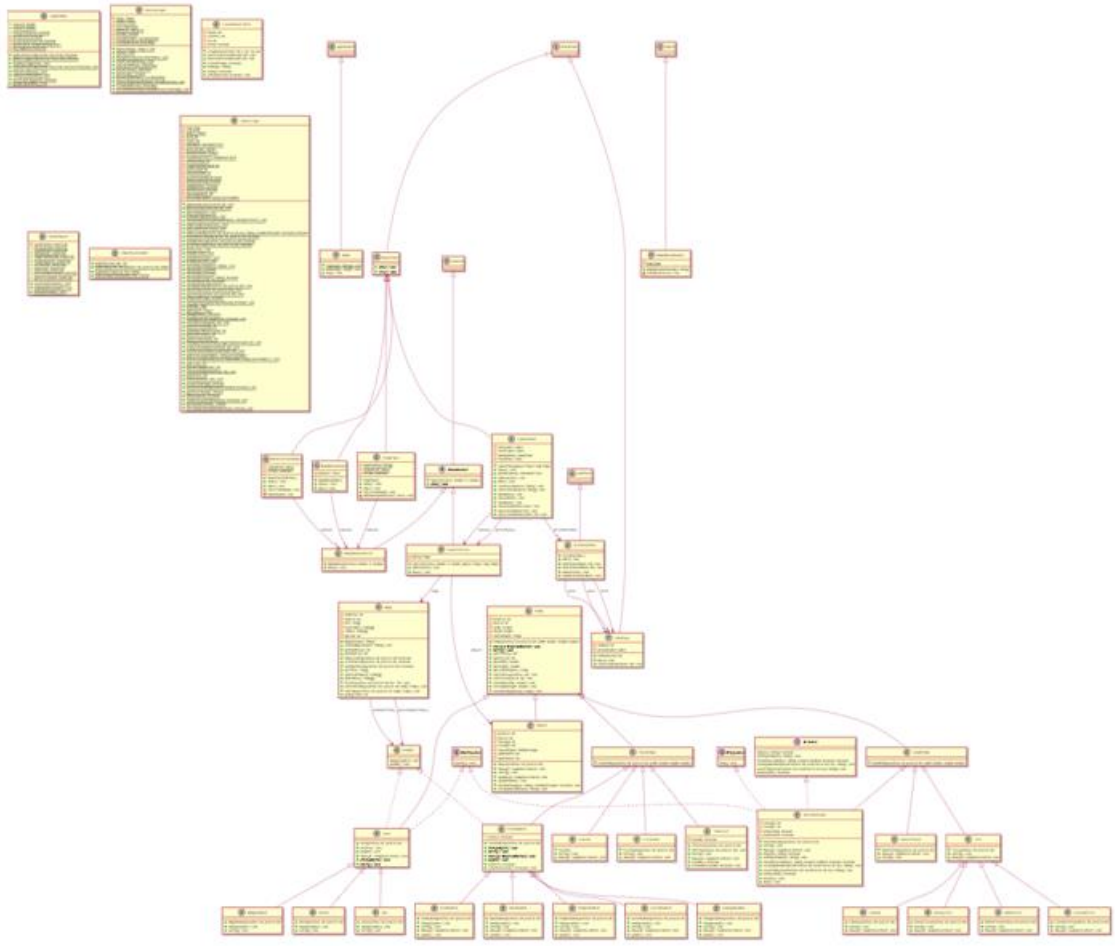
### 3.7. Time Up in Puzzle Mode

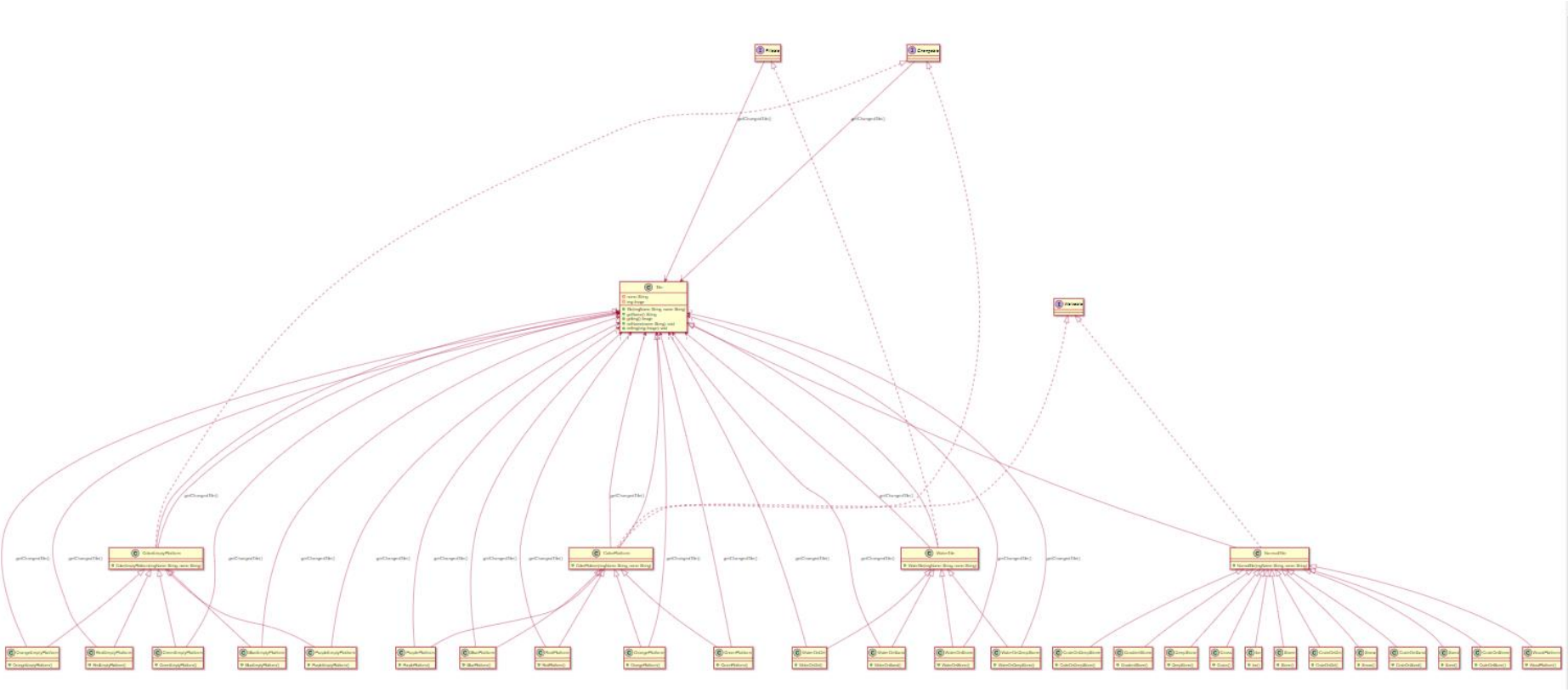
After timer is run out of time, the game over screen will appear. It will tell score that player completed. 2 buttons are available, restart the game or exit from puzzle mode.

## 4. Class Diagram

For larger picture, see the link:

https://raw.githubusercontent.com/creampiney/creampiney.github.io/master/file/progmeth/uml.svg




Flowers
Oranges
getChangedTile()
File
name: String
img: Image
File(imgName: String, name: String)
getName(): String
getImg(): Image
setName(name: String, old
changing image's nam
GiletJaune
GiletJaune(imgName: String, name: String)
GiletJauneEnemy
GiletJauneEnemy(imgName: String, name: String)
Water
Water(imgName: String, name: String)
WaterOnGrass
WaterOnGrass(imgName: String, name: String)
WaterOnGrassBoss
WaterOnGrassBoss(imgName: String, name: String)
WaterOnDirt
WaterOnDirt(imgName: String, name: String)
WaterOnDirtBoss
WaterOnDirtBoss(imgName: String, name: String)
Cake
Cake(imgName: String, name: String)
CakeEnemy
CakeEnemy(imgName: String, name: String)
CakeBoss
CakeBoss(imgName: String, name: String)
Flowers
Flowers(imgName: String, name: String)
FlowerEnemy
FlowerEnemy(imgName: String, name: String)
FlowerBoss
FlowerBoss(imgName: String, name: String)
Normal
Normal(imgName: String, name: String)
NormalEnemy
NormalEnemy(imgName: String, name: String)
NormalBoss
NormalBoss(imgName: String, name: String)
Grass
Grass(imgName: String, name: String)
GrassEnemy
GrassEnemy(imgName: String, name: String)
GrassBoss
GrassBoss(imgName: String, name: String)
Dirt
Dirt(imgName: String, name: String)
DirtEnemy
DirtEnemy(imgName: String, name: String)
DirtBoss
DirtBoss(imgName: String, name: String)
Sand
Sand(imgName: String, name: String)
SandEnemy
SandEnemy(imgName: String, name: String)
SandBoss
SandBoss(imgName: String, name: String)
Stone
Stone(imgName: String, name: String)
StoneEnemy
StoneEnemy(imgName: String, name: String)
StoneBoss
StoneBoss(imgName: String, name: String)
Ice
Ice(imgName: String, name: String)
IceEnemy
IceEnemy(imgName: String, name: String)
IceBoss
IceBoss(imgName: String, name: String)
Snow
Snow(imgName: String, name: String)
SnowEnemy
SnowEnemy(imgName: String, name: String)
SnowBoss
SnowBoss(imgName: String, name: String)
Lava
Lava(imgName: String, name: String)
LavaEnemy
LavaEnemy(imgName: String, name: String)
LavaBoss
LavaBoss(imgName: String, name: String)
Void
Void(imgName: String, name: String)
VoidEnemy
VoidEnemy(imgName: String, name: String)
VoidBoss
VoidBoss(imgName: String, name: String)

# 5. Class Details

## 5.1. Package application

### 5.1.1. Class Main extends Application

**Method**

| Name | Description |
|---|---|
| + void main(String[] args) | The entry of application |
| + void start(Stage stage) | Create application by running ScreenLogic.init(stage) |
| + void stop() | Override the method. Reset all thread and animation timer by running GameLogic.resetAnimationTimer(); |

## 5.2. Package entity.base

This package contains base entities which are interfaces and abstract classes.

### 5.2.1. Interface Actable

This interface defines methods for entities that can be interacted.

**Method**

| Name | Description |
|---|---|
| + *void changeSprite()* | This method will be called when entities are interacted. |
| + *void update()* | The method will be called in AnimationTimer to check status of actable entities. |

### 5.2.2. Interface Destroyable

This interface defines methods for entities that can be destroyed.

**Method**

| Name | Description |
|---|---|
| + *void destroy()* | This method will be called when entities are going to be destroyed. This method will remove entities from map. |

### 5.2.3. Interface Droppable

This interface defines methods for entities that can be dropped in the water (not the empty tile).

**Method**

| Name | Description |
|---|---|
| + *void drop()* | This method will be called when entities are dropped in the water. This method will change a tile in the map where the entities drop. |

### 5.2.4. Interface Slidable

This interface defines methods for entities that can be slid on ice.

**Method**

| Name | Description |
|---|---|
| + *boolean slide(String key)* | This method will be called when entities are going to slide. This method will return true if the sliding is success, otherwise return false. |
| + *void setPlayerSlip(String key)* | This method will be called to initialize the variable isPlayerSlip. The method will set the variable to true if the player is sliding, otherwise to false. |
| + *boolean checkMoveValid(String key, boolean requireCondition)* | This method will be called when entities are going to slide. This method will check for validity of |

| | |
|---|---|
| | entities' movements. If the movement is valid, the entities will slide and return true, otherwise return false. |
| *+ void movingAnimation(int newPosRow, int newPosCol, String key)* | This method will be called when entities are sliding. This method will play animation of sliding entities and called method that move entities in the map. |
| *+ void moveEntities(int newPosRow, int newPosCol, String key)* | This method will be called when animation of sliding is finished. This method will move entities on the map and check if entities are fallen in the water or continue sliding if the entities is on ice. |
| *+ boolean isAllowSide()* | This method will return isAllowSlide variable of entities. It will return true if player can slide the entities and return false when entities cannot slide or the entities is sliding. |

### 5.2.5. Abstract Class Entity

This class will represent every entity in the game and define basic methods for entities.

**Field**

| Name | Description |
|---|---|
| - int posRow | Row position in the map of entity |
| - int posCol | Column position in the map of entity |
| - double width | The width of the entity |
| - double height | The height of the entity |
| - Image currentSprite | The current image of entity which is shown in canvas. |

**Constructor**

| Name | Description |
|---|---|
| + Entity(int posRow, int posCol, double width, double height) | Initialize the fields (posRow, posCol, width, height) |

**Method**

| Name | Description |
|---|---|
| + *void draw(GraphicsContext gc)* | The method will be called when animation timer draws the canvas. |
| + *void setImg()* | The method will be called when entities is built. This will set currentSprite variable. |
| + int getPosRow() | Getter methods |
| + int getPosCol() | |
| + int getWidth() | |
| + int getHeight() | |
| + Image getCurrentSprite() | |
| + void setPosRow(int posRow) | Setter methods |
| + void setPosCol(int posCol) | |
| + void setWidth(double width) | |
| + void setHeight(double height) | |
| + void setCurrentSprite(Image img) | |

### 5.2.6. Abstract Class FloorEntity extends Entity

This class will represent the entities that can be walked through.

**Constructor**

| Name | Description |
|---|---|
| + FloorEntity(int posRow, int posCol, double width, double height) | Initialize the fields (posRow, posCol, width, height) |

### 5.2.7. Abstract Class SolidEntity extends Entity

This class will represent the solid entities that cannot be walked through.

**Constructor**

| Name | Description |
|---|---|
| + SolidEntity(int posRow, int posCol, double width, double height) | Initialize the fields (posRow, posCol, width, height) |

### 5.2.8. Abstract Class Item extends Entity implements Actable, Destroyable

This class will represent items in the game and map.

**Constructor**

| Name | Description |
|---|---|
| + Item(int posRow, int posCol) | Initialize the fields (posRow, posCol) and set width and height to 100 (px) |

**Method**

| Name | Description |
|---|---|
| + void destroy() | Set the floor entities at location of items to null.<br>(Destroy items from the map after collected) |
| + void update() | Check for player standing on the items.<br>If player is standing on the items:<br>- Change sprite of the items by calling changeSprite() |

| | |
|---|---|
| | - Add removingActable (which is the actable entity that will be remove from game) with this entity.<br>- Update amount of the items by using ScreenLogic.getInventoryPane().updateAmountLabel() |
| + void draw() | Draw image to the canvas at the items' positions in the map. |
| + *void changeSprite()* | This method will be called when player collect items. |
| + *void setImg()* | This method will be called when initialize each items. This method will set currentSprite of entity. |

## 5.3. Package entity.floor

This package contains floor entity in the game which can be walked through.

### 5.3.1. Class TileHover extends FloorEntity

This class represent opaque hovered tile which is shown when the mouse is on the tile.

**Field**

| Name | Description |
|---|---|
| - boolean isVisible | Status if the hovered tile is visible or not.<br>Has a value of true if the hovered tile is visible (The mouse is hovering on tile), otherwise has value of false. |

**Constructor**

| Name | Description |
|---|---|
| + TileHover(int posRow, int posCol) | Initialize the fields (posRow, posCol)<br>Set the height to 100 and width to 140<br>Initialize the image of entity<br>Set isVisible to false |

**Method**

| Name | Description |
|---|---|
| + void setPos(int posRow, int posCol) | Set the position of hovered tile which the mouse is hovering.<br>- If the position is outside the map, make the hover tile invisible.<br>- Else if player is using “wood” item (Can be checked by GameLogic.isMagicMode()), make the hover tile visible<br>- Else if there are tile at the mouse position, make the hover tile visible<br>- Else make the hover tile invisible |
| + void setImg() | Set the image of hovered tile. |
| + void draw(GraphicsContext gc) | Draw image to the canvas at the entity’s positions in the map. |
| + boolean isVisible() | Getter method |
| + void setVisible(boolean isVisible) | Setter method |

### 5.3.2. Class IsoGrid extends FloorEntity

This class represent isometric grid which will be shown when player is using wood item.

**Constructor**

| Name | Description |
|---|---|
| + IsoGrid() | Initialize the fields<br>- Set position to (0, 0)<br>- Set width to 1000<br>- Set height to 500<br>- Initialize the entity’s image |

Method

| Name | Description |
|---|---|
| + void setImg() | Set the image of isometric grid. |
| + void draw(GraphicsContext gc) | Draw image to the canvas at the entity's positions in the map. |

### 5.3.3. Class LevelGate extends FloorEntity

This class represent the finish gate of each level.

Constructor

| Name | Description |
|---|---|
| + LevelGate(int posRow, int posCol) | Initialize the fields (posRow, posCol)<br>Set width to 100 and height to 100<br>Set image of the entity |

Method

| Name | Description |
|---|---|
| + void setImg() | Set the image of finish gate |
| + void draw(GraphicsContext gc) | Draw image to the canvas at the entity's positions in the map. |

### 5.3.4. Abstract Class FloorButton extends FloorEntity implements Actable

This class represent button sensor that will active and inactive platforms of each color.

Field

| Name | Description |
|---|---|
| # boolean isActive | Status shows the active of button |

Constructor

| Name | Description |
|---|---|
| + FloorButton(int posRow, int posCol) | Initialize the fields (posRow, posCol)<br>Set width to 100 and height to 120<br>Set image of the entity |

Method

| Name | Description |
|---|---|
| + *void setImg()* | This method will be called when set the image of button. |
| + *void draw(GraphicsContext gc)* | This method will be called when canvas is drawing a game. Draw image to the canvas at the button's positions in the map. |
| + *void changeSprite()* | This method will be called when player or crate is standing on the button. This method will change image of sprite depend on active or inactive of button. |
| + *void update()* | This method will be called when the animation timer updates all actable entities. This method will check if player or crate is on the button or not. |
| + boolean isActive() | Getter method |
| + void setActive(boolean isActive) | Setter method |

### 5.3.5. Class RedButton extends FloorButton

This class represent red button sensor that will active and inactive red platforms.

**Constructor**

| Name | Description |
|---|---|
| + RedButton(int posRow, int posCol) | Initialize the fields (posRow, posCol) |

**Method**

| Name | Description |
|---|---|
| + void changeSprite() | Change the image of button<br>- If the red button is active, change the image to activated red button<br>- If the red button is inactive, change the image to normal red button |
| + void setImg() | This method will be called when set the image of red button. |
| + void draw(GraphicsContext gc) | Draw image to the canvas at the red button's positions in the map. |
| + void update() | Check if the player or solid entity is on the red button or not.<br>- If the player or solid entity is on red button and the button is inactive, set the button to active, change the image of button by using changeSprite(), and change all red platform from active to inactive and inactive to active.<br>- If nothing is on red button and the button is active, set the button to inactive, change |

| | |
|---|---|
| | the image of button by using changeSprite(), and change all red platform from active to inactive and inactive to active. |

### 5.3.6. Class OrangeButton extends FloorButton

This class represent orange button sensor that will active and inactive orange platforms.

**Constructor**

| Name | Description |
|---|---|
| + OrangeButton(int posRow, int posCol) | Initialize the fields (posRow, posCol) |

**Method**

| Name | Description |
|---|---|
| + void changeSprite() | Change the image of button<br>- If the orange button is active, change the image to activated orange button<br>- If the orange button is inactive, change the image to normal orange button |
| + void setImg() | This method will be called when set the image of orange button. |
| + void draw(GraphicsContext gc) | Draw image to the canvas at the orange button's positions in the map. |
| + void update() | Check if the player or solid entity is on the orange button or not.<br>- If the player or solid entity is on orange button and the button is inactive, set the button to active, change the image of |

| | |
|---|---|
| | button by using changeSprite(), and change all orange platform from active to inactive and inactive to active.<br>- If nothing is on orange button and the button is active, set the button to inactive, change the image of button by using changeSprite(), and change all orange platform from active to inactive and inactive to active. |

### 5.3.7. Class GreenButton extends FloorButton

This class represent green button sensor that will active and inactive green platforms.

**Constructor**

| Name | Description |
|---|---|
| + GreenButton(int posRow, int posCol) | Initialize the fields (posRow, posCol) |

**Method**

| Name | Description |
|---|---|
| + void changeSprite() | Change the image of button<br>- If the green button is active, change the image to activated green button<br>- If the green button is inactive, change the image to normal green button |
| + void setImg() | This method will be called when set the image of green button. |
| + void draw(GraphicsContext gc) | Draw image to the canvas at the green button's positions in the map. |

| | |
|---|---|
| + void update() | Check if the player or solid entity is on the green button or not.<br>- If the player or solid entity is on green button and the button is inactive, set the button to active, change the image of button by using changeSprite(), and change all green platform from active to inactive and inactive to active.<br>- If nothing is on green button and the button is active, set the button to inactive, change the image of button by using changeSprite(), and change all green platform from active to inactive and inactive to active. |

### 5.3.8. Class BlueButton extends FloorButton

This class represent blue button sensor that will active and inactive blue platforms.

**Constructor**

| Name | Description |
|---|---|
| + BlueButton(int posRow, int posCol) | Initialize the fields (posRow, posCol) |

**Method**

| Name | Description |
|---|---|
| + void changeSprite() | Change the image of button<br>- If the blue button is active, change the image to activated blue button<br>- If the blue button is inactive, change the image to normal blue button |

| | |
|---|---|
| + void setImg() | This method will be called when set the image of blue button. |
| + void draw(GraphicsContext gc) | Draw image to the canvas at the blue button's positions in the map. |
| + void update() | Check if the player or solid entity is on the blue button or not.<br>- If the player or solid entity is on blue button and the button is inactive, set the button to active, change the image of button by using changeSprite(), and change all blue platform from active to inactive and inactive to active.<br>- If nothing is on blue button and the button is active, set the button to inactive, change the image of button by using changeSprite(), and change all blue platform from active to inactive and inactive to active. |

### 5.3.9. Class PurpleButton extends FloorButton

This class represent purple button sensor that will active and inactive purple platforms.

**Constructor**

| Name | Description |
|---|---|
| + PurpleButton(int posRow, int posCol) | Initialize the fields (posRow, posCol) |

**Method**

| Name | Description |
|---|---|
| + void changeSprite() | Change the image of button |

| | |
|---|---|
| | - If the purple button is active, change the image to activated purple button<br>- If the purple button is inactive, change the image to normal purple button |
| + void setImg() | This method will be called when set the image of purple button. |
| + void draw(GraphicsContext gc) | Draw image to the canvas at the purple button's positions in the map. |
| + void update() | Check if the player or solid entity is on the purple button or not.<br>- If the player or solid entity is on purple button and the button is inactive, set the button to active, change the image of button by using changeSprite(), and change all purple platform from active to inactive and inactive to active.<br>- If nothing is on purple button and the button is active, set the button to inactive, change the image of button by using changeSprite(), and change all purple platform from active to inactive and inactive to active. |

## 5.4. Package entity.items

This package contains items entity in the game. There are 3 kinds of items.

### 5.4.1. Class MagicWand extends Item

This class represent magic wand items which dropped on the floor.

**Constructor**

| Name | Description |
|---|---|
| + MagicWand(int posRow, int posCol) | Initialize the fields (posRow, posCol) |

**Method**

| Name | Description |
|---|---|
| + void changeSprite() | Collect the magic wand from the floor.<br>- Add amount of magic wand by 1 by using GameLogic.setMagicWandAmount(amount)<br>- Destroy the entity |
| + void setImg() | Set the image of magic wand entity. |

### 5.4.2. Class Axe extends Item

This class represent axe items which dropped on the floor.

**Constructor**

| Name | Description |
|---|---|
| + Axe(int posRow, int posCol) | Initialize the fields (posRow, posCol) |

**Method**

| Name | Description |
|---|---|
| + void changeSprite() | Collect the axe from the floor.<br>- Add amount of axe by 1 by using GameLogic.setAxeAmount(amount)<br>- Destroy the entity |
| + void setImg() | Set the image of axe entity. |

### 5.4.3. Class Wood extends Item

This class represent wood items which dropped on the floor.

**Constructor**

| Name | Description |
|---|---|
| + Wood(int posRow, int posCol) | Initialize the fields (posRow, posCol) |

**Method**

| Name | Description |
|---|---|
| + void changeSprite() | Collect the wood from the floor.<br>- Add amount of axe by 1 by using GameLogic.seWoodAmount(amount)<br>- Destroy the entity |
| + void setImg() | Set the image of wood entity. |

## 5.5. Package entity.player

This package contains player entity which is main character of game.

### 5.5.1. Class Player extends Entity

This class represent player entity which player can control.

**Field**

| Name | Description |
|---|---|
| - int posRow | Row position in the map of entity |
| - int posCol | Column position in the map of entity |
| - WritableImage currentSprite | The current image of entity which is shown in canvas. |
| - int movingX | X position of animated player from original position |
| - int movingY | Y position of animated player from original position |

| | |
|---|---|
| - int spriteNumX | X index of currentSprite in the player's whole image. |
| - int spriteNumY | Y index of currentSprite in the player's whole image. |

Constructor

| Name | Description |
|---|---|
| + Player(int posRow, int posCol) | - Initialize the fields (posRow, posCol)<br>- Set the width and height to 100<br>- Set spriteNumX, spriteNumY, movingX, movingY to 0<br>- Update sprite the entity<br>- Set isPlayerMoving in GameLogic to false |

Method

| Name | Description |
|---|---|
| + void draw(GraphicsContext gc) | Draw image on the canvas at the player's position |
| + void setImg() | Update sprite of entity |
| + void update(GraphicsContext gc) | Check if user press button or left click the mouse.<br>- If player is not moving and game is still running, allow user to check the button triggered.<br>- If triggered button is W A S or D, check the moving at the direction W A S or D.<br>- If triggered button is R, restart the level.<br>- If triggered button is 1-3, choose inventory slot 1-3. |

| | |
|---|---|
| | - If triggered button is ESCAPE, open menu pane or close pane depend on if menu is open or not.<br>- If left click is triggered, check for items using.<br>- If user is choosing items, called GameLogic method that consume items depends on items user choose. |
| + void updateSprite() | Set image of sprite to new index of player's whole image.<br>Increase spriteNumX by 1<br>If spriteNumX is 20, change it to 0 |
| + void checkMoving(String key, boolean requireCondition) | Check if key is equals to W A S or D, and check condition for moving player. If the condition is meet or requireCondition is false, called movingAnimation.<br>Change the spriteNumY depends on direction or key. |
| + void movingAnimation(String key) | Set player moving to true.<br>Play player moving sound.<br>Create thread which represent animation of moving player depends on key and start the thread.<br>After thread end, check player status by using GameLogic.checkPlayerStat(key); |

### 5.6. Package entity.solid

This package contains solid entity which player can't pass through.

### 5.6.1. Class WoodenCrate extends SolidEntity implements Slidable, Destroyable, Droppable

This class represent wooden crate which player can push. The feature of this crate is droppable in water.

**Field**

| Name | Description |
|---|---|
| - int movingX | X position of animated entity from original position |
| - int movingY | Y position of animated entity from original position |
| - boolean isPlayerSlip | Boolean describe if player is slipping or not |
| - boolean isAllowSlide | Boolean describe if crate can slide or not |

**Constructor**

| Name | Description |
|---|---|
| + WoodenCrate(int posRow, int posCol) | - Initialize the fields (posRow, posCol)<br>- Set the width to 100 and height to 110<br>- Set isAllowSlide to true |

**Method**

| Name | Description |
|---|---|
| + void setImg() | Set the image of wooden crate entity. |
| + void draw(GraphicsContext gc) | Draw image of wooden crate at crate's position. |
| + boolean slide(String key) | Check if crate can slide or not, if the crate can slide, set isPlayerSlip on the condition in setPlayerSlip(key) and check for move valid and return boolean from this moveValid method. |
| + void setPlayerSlip(String key) | Check if the next tile player is going to move is ice or not. If it is ice, set isPlayerSlip to true, else set to false. |

| | |
|---|---|
| + boolean checkMoveValid(String key, boolean requireCondition) | Check for condition of next position crate going to move. Return true if success, otherwise return false.<br>- If there is tile in the next position or not outside map or not require condition, allow to move crate, and return true<br>- If there is entity at the next position or no tiles at next position and the block allow side, not allow to move crate, return false.<br>- Else if the next position is outside map and next position's tile is not walkable, not allow to move crate, return false.<br>- Otherwise, allow moving crate, return true. |
| + void movingAnimation(int newPosRow, int newPosCol, String key) | Crate thread for crate's moving animation to new position for each direction. After thread end, move the entity on the map. |
| + void moveEntity (int newPosRow, int newPosCol, String key) | Move the entity in the map.<br>- If the new position is not outside map, move this crate to new position.<br>Set the entity at old position to null.<br>Set row and column position to new one.<br>Check for condition after sliding<br>- If the tile under crate is Fillable, drop the crate.<br>- If the tile under crate is not walkable, destroy the crate.<br>- If the tile under crate is ice and player is not slip, continue slide the crate by calling checkMoveValid(key, false) again. |
| + void destroy() | Set the entity at the current position to null. |

| | |
|---|---|
| + void drop() | Create thread that wait for 150 ms until the crate drop and start the thread.<br>Destroy the crate at new position.<br>Change the tile at dropped position to filled tile. |
| + boolean isAllowSlide() | Getter method |

### 5.6.2. Class Tree extends SolidEntity

This class represent tree which player cannot pass through, but player can chop the tree.

**Constructor**

| Name | Description |
|---|---|
| + Tree(int posRow, int posCol) | - Initialize the fields (posRow, posCol)<br>- Set the width to 100 and height to 120<br>- Set the image by using setImg() |

**Method**

| Name | Description |
|---|---|
| + void setImg() | Set the image of tree entity. |
| + void draw(GraphicsContext gc) | Draw image of tree at tree's position. |

### 5.6.3. Class SnowyTree extends Tree

This class represent snowy tree which is a kind of tree.

**Constructor**

| Name | Description |
|---|---|
| + SnowyTree(int posRow, int posCol) | - Initialize the fields (posRow, posCol)<br>- Set the image by using setImg() |

**Method**

| Name | Description |
|---|---|
| + void setImg() | Set the image of snowy tree entity. |
| + void draw(GraphicsContext gc) | Draw image of snowy tree at snowy tree's position. |

### 5.6.4. Class CoconutTree extends Tree

This class represent coconut tree which is a kind of tree.

**Constructor**

| Name | Description |
|---|---|
| + CoconutTree(int posRow, int posCol) | - Initialize the fields (posRow, posCol)<br>- Set the image by using setImg() |

**Method**

| Name | Description |
|---|---|
| + void setImg() | Set the image of coconut tree entity. |
| + void draw(GraphicsContext gc) | Draw image of coconut tree at snowy tree's position. |

### 5.6.5. Class Cactus extends Tree

This class represent cactus which is a kind of tree.

**Constructor**

| Name | Description |
|---|---|
| + Cactus(int posRow, int posCol) | - Initialize the fields (posRow, posCol)<br>- Set the image by using setImg() |

**Method**

| Name | Description |
|---|---|
| + void setImg() | Set the image of cactus entity. |
| + void draw(GraphicsContext gc) | Draw image of cactus at cactus's position. |

### 5.6.6. Class WinterTree extends Tree

This class represent winter tree which is a kind of tree.

**Constructor**

| Name | Description |
|---|---|
| + WinterTree(int posRow, int posCol) | - Initialize the fields (posRow, posCol)<br>- Set the image by using setImg() |

**Method**

| Name | Description |
|---|---|
| + void setImg() | Set the image of winter tree entity. |
| + void draw(GraphicsContext gc) | Draw image of winter tree at winter tree's position. |

### 5.6.7. Class SlimeFriend extends SolidEntity

This class represent slime friend which is decoration in last level of story mode.

**Constructor**

| Name | Description |
|---|---|
| + SlimeFriend(int posRow, int posCol) | - Initialize the fields (posRow, posCol)<br>- Set width and height to 100<br>- Set the image by using setImg() |

Method

| Name | Description |
|---|---|
| + void setImg() | Set the image of slime friend entity. |
| + void draw(GraphicsContext gc) | Draw image of slime friend at it's position. |

### 5.7. Package input

This package contains input utility.

#### 5.7.1. Class InputUtility

This class will check for mouse and keyboard input.

Field

| Name | Description |
|---|---|
| + double mouseX | X position of mouse |
| + double mouseY | Y position of mouse |
| - boolean isLeftDown | Boolean describe if left mouse is clicked or not |
| - boolean isLeftClickedLastTick | Boolean describe if left mouse last tick |
| - ArrayList<KeyCode> keyPressed | ArrayList that contains pressed key |
| - KeyCode keyTriggered | KeyCode that contains triggered key |

Method

| Name | Description |
|---|---|
| + boolean boolean getKeyPressed(KeyCode keycode) | If the keycode is pressed, return true, otherwise return false. |
| + boolean getKeyTriggered(KeyCode keycode) | If the keycode is triggered, return true, otherwise return false. |
| + void clearKeyTriggered() | Set keyTriggered to null |
| + void setKeyPressed(KeyCode keycode,boolean pressed) | If the button is pressed, check the condition: |

| | |
|---|---|
| | - If keycode not contains in the keyPressed, add it and set keyTriggered to that keycode.<br>If the button is released, remove keycord from keyPressed and clear keyTriggered. |
| + void mouseLeftDown() | Called when left mouse is down. |
| + void mouseLeftRelease() | Release the left mouse |
| + boolean isLeftClickTriggered() | If the left mouse is triggered, return true, otherwise return false. |
| + void updateInputState() | Clear triggered key and clear left click triggered. |

### 5.8. Package logic

This package contains logic of the game, audio player, and object generator.

#### 5.8.1. Class AudioPlayer

This class represent the audio player. Every audio will be play here.

**Field**

| Name | Description |
|---|---|
| - AudioClip audioRunner | The audio which is used to play background sound in menu and game. |
| + AudioClip movingAudio | The audio which will be play when player is moving. |
| + AudioClip fallingAudio | The audio which will be play when player is falling. |
| + AudioClip magicWandAudio | The audio which will be play when player is using magic wand. |
| + AudioClip choppingAudio | The audio which will be play when player is chopping tree. |
| + AudioClip woodAudio | The audio which will be play when player is placing wood platform. |

| | |
|---|---|
| + AudioClip clickAudio | The audio which will be play when user is clicking button. |
| + AudioClip levelCompleteAudio | The audio which will be play when player complete level. |
| + AudioClip gameOverAudio | The audio which will be play when time is up in puzzle mode. |

**Method**

| Name | Description |
|---|---|
| + void resetAudioRunner() | Set audioRunner to null |
| + void runMainMenuAudio() | Reset audio runner and play main menu background sound with indefinite loop. |
| + void runGameAudio() | Reset audio runner and play game background sound with indefinite loop. |

### 5.8.2. Class CountdownTimer

This class represent the timer which is used in puzzle mode.

**Field**

| Name | Description |
|---|---|
| - int minute | Minute part of timer |
| - int second | Second part of timer |
| - int ms | Millisecond part of timer |
| - boolean minute | Boolean describe if the timer is stop or not |

**Constructor**

| Name | Description |
|---|---|
| + CountdownTimer(int m, int s, int ms) | Initialize the field (m, s, ms) |

| | |
|---|---|
| | Set isStop to true |

**Method**

| Name | Description |
|---|---|
| + void decrementTimer(int amount) | Decrease the timer by specified amount (milliseconds) |
| + incrementTimer(int amount) | Increase the timer by specified amount (seconds) |
| + boolean isTimerEmpty() | Return true if timer is empty (time is less than or equal 0) |
| + String toString() | Return string format of timer in format of 00:00:00 |
| + boolean isStop() | Getter method |
| + void setStop(boolean isStop) | Setter method |

### 5.8.3. Class GameLogic

This class represent every logic about game.

**Field**

| Name | Description |
|---|---|
| - Map map | The map in the game |
| - Player player | The player in the game |
| - int level | The level which the game is running |
| - int score | The score obtained in puzzle mode |
| - AnimationTimer animation | The animation timer use to draw game canvas and update input |
| - Thread timerThread | The thread represent timer in puzzle mode |
| - Thread tutorialThread | The thread represent tutorial panel which is shown in story mode. |
| - CountdownTimer countdownTimer | The countdown timer which is used in puzzle mode |
| - int slotSelecting | The selected slot of inventory |

| | |
|---|---|
| | 0 for nothing<br>1 for magic wand<br>2 for axe<br>3 for wood |
| - int magicWandAmount | The amount of magic wand left in the level |
| - int axeAmount | The amount of axe left in the level |
| - int woodAmount | The amount of wood left in the level |
| - boolean isGameRunning | Boolean describing is the game running or not |
| - boolean isPlayerMoving | Boolean describing is the player moving or not |
| - boolean isMagicMode | Boolean describing is the isometric grid is shown or not (Shown when player is choosing wood item) |
| - boolean isMenuOpen | Boolean describing is the menu pane in the game is shown or not |
| - int gameplayMode | Number indicating mode of game<br>1 for Story Mode<br>2 for Puzzle Mode |
| - ArrayList<Actable> removingActable | Array list that store actable entity that require to remove from updating array list in animation timer. |

Method

| Name | Description |
|---|---|
| + void initNormalGame(int levelInt) | Create new story mode game.<br>Set every field to initial and clear all thread.<br>Build map with specific level, player, and change scene of stage. After build successfully, check for tutorial pane in each level. |
| + void initTimerGame(int levelInt) | Create new puzzle mode game.<br>Set every field to initial and clear all thread. |

| | |
|---|---|
| | Build map with specific level, player, and change scene of stage. After build successfully, start the timer |
| + void initTimerMode() | This will be called if player enter puzzle mode. Initialize timer to 1 minute and 30 seconds and set score to -1 and call nextLevel().<br>(In the method nextLevel, time in timer will be added by 30 second and score will be added by 1. Therefore, initial time and score is 2 minutes and 0 respectively) |
| + void resetAnimationTimer() | Clear all animation timer and thread. |
| + void setAnimationTimer(AnimationTimer newAnimation) | Set animation timer to specific animation timer. Start the animation timer. |
| + void startCountDownTimer() | Create thread that start timer, and start the thread. If the timer got InterruptedException, stop the timer. |
| + void runCountDownTimer() throws InterruptedException | Start the timer. If the timer is run out of time, stop the timer and end the puzzle mode game. |
| + boolean isMoveValid(int posRow, int posCol, String key, boolean conditionRequire) | Check if the player movement is allowed or not<br>- If the position is outside the map, not allow to move.<br>- If the entity is placed at the position and that entity is slideable, check that if that entity is valid to move and allow sliding, try to slide the entity, return true if it is success, otherwise reture false.<br>- If the entity is placed at the position and that entity is solid, not allow to move.<br>- Otherwise, check that is the tile at that position is walkable or not. |

| | |
|---|---|
| + boolean isSolidMoveValid(int posRow, int posCol) | Return true if solid entity can move to that position, otherwise return false. |
| + boolean isEntityPlaced(int posRow, int posCol) | Return true if entity is placed in that position, otherwise return false. |
| + boolean isTilePlaced(int posRow, int posCol) | Return true if tile at that position is not empty, otherwise return false. |
| + void nextLevel() | Proceed next level of the game, play level complete sound.<br>If story mode is running, check for this condition:<br>- If the next level is level 51, story mode end.<br>- Otherwise, create next level<br>If puzzle mode is running, add the score by 1 and crate next RANDOM level, |
| + void restartLevel() | Create the same level for each mode. |
| + void storyModeClear() | This will be called when complete level 50 of story mode. Set game running to false and show congratulation pane. |
| + void timerModeEnd() | This will be called when time is up in puzzle mode. Set game running to false, play the game over audio and show time up pane. |
| + void checkPlayerStat(String key) | Check status of player after moving.<br>- If player fall, play fall audio and restart the level.<br>- If player win, create next level.<br>- If player is done slipping while pushing crate, set player moving to false.<br>- If player slip on ice, continue move player in the same direction.<br>- Otherwise, set player moving to false. |

| | |
|---|---|
| + boolean checkWin() | Return true if floor entity at player's position is finish gate, otherwise return false. |
| + boolean checkSlip() | Return true if tile at player's position is ice, otherwise return false. |
| + boolean checkSlipDone(String key) | Check for the next position of player. Return true if one of the following conditions is true:<br>- Check for solid entity (but not slidable) collide with player<br>- Check for slidable entity collides with solid entity<br>- Check for slidable entity dropping<br>Otherwise, return false. |
| + boolean checkPlayerFall() | Return true if player is outside map or tile at player's position is not walkable, otherwise return false. |
| + void useMagicWand(int posRow, int posCol) | This will be called when player use magic wand.<br>Decrease magic wand amount by 1 and move player to specific position.<br>Play magic wand audio. |
| + void useAxe(int posRow, int posCol) | This will be called when player use axe.<br>Decrease axe amount by 1 and set entity at that position to null.<br>Play chopping audio. |
| + void useWood(int posRow, int posCol) | This will be called when player use wood.<br>Decrease wood amount by 1 and set tile at specific position to wooden platform.<br>Play wood audio. |
| + void setPlayerMoving(boolean isPlayerMoving) | Setter Method |
| + void setMagicMode(boolean magicMode) | |

| | |
|---|---|
| + void setSlotSelecting(int slot) | Setter Method |
| + void setMagicWandAmount(int magicWandAmount) | |
| + void setAxeAmount(int axeAmount) | |
| + void setWoodAmount(int woodAmount) | |
| + void setRemovingActable(ArrayList<Actable> removingActable) | |
| + void setGameplayMode(int mode) | |
| + void setScore(int score) | |
| + void setGameRunning(boolean isGameRunning) | |
| + void setMenuOpen(boolean isMenuOpen) | |
| + void setTutorialThread(Thread tutorialThread) | |
| + boolean isPlayerMoving() | Getter Method |
| + Map getMap() | |
| + Player getPlayer() | |
| + boolean isMagicMode() | |
| + int getSlotSelecting() | |
| + int getMagicWandAmount() | |
| + int getAxeAmount() | |
| + int getWoodAmount() | |
| + ArrayList<Actable> getRemovingActable() | |
| + int getLevel() | |
| + int getGameplayMode() | |
| + int getScore() | |

| | |
|---|---|
| + boolean isGameRunning() | Getter Method |
| + Thread getTimerThread() | |
| + boolean isMenuOpen() | |
| + Thread getTutorialThread() | |

### 5.8.4. Class Map

This class represent map in each level. This class contains tiles, floor entity, entity of each level.

**Field**

| Name | Description |
|---|---|
| - int startRow | Row position which player start |
| - int startCol | Column position which player start |
| - Tile[][] tiles | Array of tile in each position |
| - Entity[][] floorEntities | Array of floor entity in each position |
| - Entity[][] entities | Array of entity in each position |
| - int bgCode | Background code for each level |
| - ArrayList<Actable> actableEntity | Array list of actable entity need to update |

**Constructor**

| Name | Description |
|---|---|
| + Map(String filename) | Initialize all arrays with 10x10 size<br>Initialize actableEntity with new ArrayList<Actable>()<br>Create map with file name |

Method

| Name | Description |
|---|---|
| + void createMap(String filename) | Read file from resource folder and import start position, tiles, floor entities, entities, background code into field. Use : ObjectGenerator.buildTile(code) to generate tiles ObjectGenerator.buildEntity(code) to generate floor entity and entity. |
| + boolean isMoveValid(int posRow, int posCol) | Return true if tiles at this position is walkable, otherwise return false. |
| + boolean isTilePlaced(int posRow, int posCol) | Return true if tiles at this position is not null, otherwise return false. |
| + boolean isEntityPlaced(int posRow, int posCol) | Return true if entities at this position is not null, otherwise return false. |
| + void setTile(int posRow, int posCol, Tile tile) | Set tile at specific position to specific tile. |
| + void setFloorEntity(int posRow, int posCol, Entity entity) | Set floor entity at specific position to specific entity. |
| + void setEntity(int posRow, int posCol, Entity entity) | Set entity at specific position to specific entity. |
| + int getStartRow() | Getter Method |
| + int getStartCol() | |
| + Tile[][] getTiles() | |
| + Entity[][] getFloorEntities() | |
| + Entity[][] getEntities() | |
| + ArrayList<Actable> getActableEntity() | |
| + int getBgCode() | |

### 5.8.5. Class ObjectGenerator

This class will generate tiles, entities, story text, and tutorial thread.

**Method**

| Name | Description |
|---|---|
| + Tile buildTile(int code) | Return tile which meet the specific code. |
| + Entity buildEntity(int code, int posRow, int posCol) | Return entity which meet the specific code. |
| + String buildStoryText(int level) | Return story string of each level in story mode. |
| + Thread buildTutorialThread(int index) | Return thread that show tutorial pane for each tutorial. |

### 5.8.6. Class ScreenLogic

This class will control everything about screen such as changing screen.

**Field**

| Name | Description |
|---|---|
| - Stage stage | Stage of the application |
| - Scene scene | The scene which is shown in the stage |
| - BasePane root | The current pane which is showing |
| - TileHover tileHover | Tile hover entity which is visible when player hovers on the tile |
| - final IsoGrid isoGrid | The isometric grid which is shown when player is selecting wood in inventory |
| - InventoryPane inventoryPane | The pane which can be selected items at the bottom right of the scene |
| - boolean[] isTutorialShown | Array of boolean which tell that tutorial pane had ever been shown or not |

Method

| Name | Description |
|---|---|
| + void init(Stage newStage) | Initialize stage, scene, and show main menu pane for first time. Play main menu audio. |
| + void show() | Show stage of application |
| + void changeScene(BasePane root) | Change the root as specific base pane, request focus for new root and initialize tileHover. |
| + void checkTutorialShow() | Check if the tutorial has ever been shown in each level or not. If it hasn't been shown, generate new animation thread and show it, and set isTutorialShown of that index to false. |
| + BasePane getCurrentPane() | Getter Method |
| + TileHover getTileHover() | |
| + IsoGrid getIsoGrid() | |
| + InventoryPane getInventoryPane() | |
| + boolean[] isTutorialShown() | |
| + void setInventoryPane(InventoryPane invPane) | Setter Method |
| + void setTutorialShown(boolean[] isTutorialShown) | |

## 5.9. Package screen

This package contains every pane and screen of the game including main menu, game, and subpanel.

### 5.9.1. Abstract Class BasePane extends StackPane

This abstract class is the base of every root that will be shown on scene.

**Method**

| Name | Description |
|---|---|
| + *void draw()* | This method will be called when the pane is initialized and draw canvas. |
| + *void initUI()* | This method will be called when the pane is initialized. This will build UI for each base pane. |

### 5.9.2. Abstract Class BaseScreen extends Canvas

This abstract class is the base of every canvas in base screen.

**Constructor**

| Name | Description |
|---|---|
| + BaseScreen(double w, double h) | Initialize the canvas. |

**Method**

| Name | Description |
|---|---|
| + *void draw()* | This method will be called when the pane is requested to draw. This method will draw background and graphics context. |

### 5.9.3. Class GamePane extends BasePane

This class is the root of game pane in story mode and puzzle mode.

**Field**

| Name | Description |
|---|---|
| - GameScreen canvas | The canvas which will be used to draw a game. |
| - Label storyLabel | The label showing story in the story mode or showing timer in puzzle mode. |

| | |
|---|---|
| - Label sceneLabel | The label showing scene number in the story mode or showing score in puzzle mode. |
| - StackPane allStackPane | The pane which include all supplementary panes in the game pane. |
| - InventoryPane inventoryPane | The pane that is at bottom right of the game pane. User can choose items in this pane. |
| - VBox menuPane | The pane including restart button and menu button at top right of the game pane. |

Constructor

| Name | Description |
|---|---|
| + GamePane(Player player, Map map) | Initialize the GameScreen canvas with width of 1280 and height of 800.<br>Add this canvas into this pane.<br>Set animation timer of this pane in GameLogic.<br>Add listener, initialize UI and build menu and inventory bar.<br>Set the text of story label and scene label<br>- If story mode is running, story label will be obtained from ObjectGenerator.buildStoryText and show scene label in form of “Scene <level> / 50”<br>- If puzzle mode is running, story label will be timer from CountdownTimer in GameLogic and scene label will be score which can be obtained from GameLogic |

Method

| Name | Description |
| --- | --- |
| + void draw() | Draw the canvas |
| + AnimationTimer getAnimation() | Return animation timer which run following steps:<br>- Draw the canvas<br>- Update input<br>- Check for magic mode which will be shown when player is selecting wood items<br>- Clear removingActable in GameLogic<br>- Update each actable entity which is in map<br>- After run these step 1 time, remove actable entities which are in removingActable in GameLogic |
| + void addListener() | Add event listener of the pane:<br>- If keyboard pressed, add key pressed in InputUtility<br>- If keyboard released, remove key pressed in InputUtility<br>- If mouse pressed, call mouse left down in InputUtility<br>- If mouse released, call mouse left release in InputUtility<br>- If mouse is in pane, set mouse on screen to true in InputUtility<br>- If mouse exit from pane, set mouse on screen to false in InputUtility<br>- If mouse is moved, check position where mouse is located, if the mouse is on the tiles, set position of tile hover in ScreenLogic to that position. |

| | |
|---|---|
| + void initUI() | This method will Initialize UI of this pane including story label, scene label, inventory pane, menu bar.<br>Set event listener of restart and menu button<br>- If mouse is over the buttons, change background of button<br>- If the button is clicked<br>Restart Button: restart level and play clicked audio.<br>Menu Button: show menu pane, stop game running, and play clicked audio |
| + void setStoryLabel(String text) | Set text of story label |
| + void setSceneLabel(String text) | Set text of scene label |
| + void buildMenu() | Show menu pane which include resume button, restart button, and exit button.<br>- If resume button is clicked, continue playing game<br>- If restart button is clicked, restart that level for story mode and create new game for puzzle mode<br>- If exit button is clicked, clear every animation timer and thread, and show main menu pane with main menu audio. |
| + void showMenu() | If puzzle mode is running, stop countdown timer.<br>Add menu pane in this pane and change setMenuOpen in GameLogic to true. |
| + void hideMenu() | If puzzle mode is running, resume countdown timer.<br>Remove menu pane in this pane and change setMenuOpen in GameLogic to false. |
| + void showStoryModeClear() | Show the congratulation pane when story mode is cleared including restart and exit button. |

| | |
|---|---|
| | - If restart button is clicked, create new game for story mode.<br>- If exit button is clicked, clear every animation timer and thread, and show main menu pane with main menu audio. |
| + void showTimerModeEnd() | Show the game over pane when time is up in puzzle mode including restart and exit button.<br>- If restart button is clicked, create new game for puzzle mode.<br>- If exit button is clicked, clear every animation timer and thread, and show main menu pane with main menu audio. |
| + void showTutorialPane(int index) | Show tutorial pane which tell the tutorial of level with resume button. |

### 5.9.4. Class GameScreen extends BaseScreen

This class is the canvas which will be drawn in GamePane.

**Field**

| Name | Description |
|---|---|
| - Map map | The map which will be drawn |
| - Player player | The player which will be drawn |
| - Image bgImg | The background of the canvas |

**Constructor**

| Name | Description |
|---|---|
| + GameScreen(double w, double h, Player player, Map map) | Initialize the base screen and field.<br>Add listener to the screen |

Method

| Name | Description |
| --- | --- |
| + void addListener() | Add event listener of the screen:<br>- If keyboard pressed, add key pressed in InputUtility<br>- If keyboard released, remove key pressed in InputUtility<br>- If mouse pressed, call mouse left down in InputUtility<br>- If mouse released, call mouse left release in InputUtility<br>- If mouse is in pane, set mouse on screen to true in InputUtility<br>- If mouse exit from pane, set mouse on screen to false in InputUtility |
| + void draw() | Draw the game in graphic contexts with the order from back to front:<br>- Draw tiles of the map<br>- Draw tile hover if tile hover is visible<br>- Draw floor entities of the map<br>- If magic mode is active, draw isometric grid<br>- Draw player if player is in this position<br>- Draw entity of the map |

### 5.9.5. Class HelpPane extends BaseScreen

This class is the root of the scene where all help and tutorial will be shown.

**Field**

| Name | Description |
|---|---|
| - final String[] buttonString | Button string of all tutorial selectors. |
| - final String defaultFont | The default font of the text in label and button. |
| - MainMenuScreen canvas | The canvas of the pane. |
| - GridPane UIPane; | Pane which show tutorial and selectors. |

**Constructor**

| Name | Description |
|---|---|
| + HelpPane() | Initialize main menu screen and add this canvas into this pane.<br>Draw this canvas and initialize UI. |

**Method**

| Name | Description |
|---|---|
| + void draw() | Draw the canvas. |
| + void initUI() | Initialize UI of the pane including tutorial selectors at the left and tutorial image at the right.<br>Each button in selectors will be initialize using initButton in this class. |
| + void setUIConstraint() | Set constraint of the UI pane in each row and column. |
| + void initButton(VBox buttonVBox) | Initialize button for each buttonVBox with mosue clicked event |

| | |
|---|---|
| | - If the button is clicked, play button clicked audio, remove old tutorial image and show new one. |

### 5.9.6. Class InventoryPane extends GridPane

This class is the inventory pane which is shown in game pane.

**Field**

| Name | Description |
|---|---|
| - SlotPane slot1 | Item slot 1 |
| - SlotPane slot2 | Item slot 2 |
| - SlotPane slot3 | Item slot 3 |

**Constructor**

| Name | Description |
|---|---|
| + InventoryPane() | Initialize UI of the pane. |

**Method**

| Name | Description |
|---|---|
| + void initUI() | Create UI of the pane including 3 slot panes.<br>Set on mouse clicked event, if the slot is clicked, called clickSlot of each slot.<br>Update amount label of each slot panes. |
| + void clickSlot(int slotNum) | If player is selecting this slot, set this slot to inactive.<br>Otherwise, set other slot to inactive and active this slot. |

| | |
|---|---|
| + void activeSlot(int slotNum) | For item that player choose, if amount of the item is 0, stop next process.<br>Set slotSelecting in GameLogic to this slot number.<br>Set border of this slot to be different from other slot. |
| + void inactiveSlot() | Set slotSelecting in GameLogic to 0.<br>Set border of all slot to original one. |
| + void updateAmountLabel() | Set amount label of all slots to be same as amount in GameLogic. |

### 5.9.7. Class MainMenuButton extends Button

This class is the button which is used in almost every pane.

**Field**

| Name | Description |
|---|---|
| - final Font font | Default font of the button |

**Constructor**

| Name | Description |
|---|---|
| + MainMenuButton(String label) | Initialize button with this label.<br>Set width to 300 and height to 50.<br>Set padding, style, and font of button<br>Set button hover event handling. |

**Method**

| Name | Description |
|---|---|
| - void setButtonHover() | Set mouse hovering event<br>- If mouse is over the button, change the style to be opaquer. |

| | |
|---|---|
| | - If mouse is exit, change style to original one. |

### 5.9.8. Class MainMenuPane extends BasePane

This class is the main menu pane which is shown when game start.

**Field**

| Name | Description |
|---|---|
| - MainMenuScreen canvas | Canvas of this pane |
| - VBox buttonUI | VBox that consists of all buttons |

**Constructor**

| Name | Description |
|---|---|
| + MainMenuPane() | Initialize main menu canvas and add this canvas to this pane.<br>Draw the canvas and initialize UI. |

**Method**

| Name | Description |
|---|---|
| + void initUI() | Initialize UI of the pane including game topic, start, help, exit buttons.<br>Set on mouse click event handling of each button:<br>- If start button is clicked, change scene to mode selector pane<br>- If help button is clicked, change scene to help pane<br>- If exit button is clicked, exit the program |

### 5.9.9. Class MainMenuScreen extends BaseScreen

This class is the canvas which is shown in all pane in main menu.

**Constructor**

| Name | Description |
|---|---|
| + MainMenuScreen() | Initialize base screen |

**Method**

| Name | Description |
|---|---|
| + void draw() | Draw the background of main menu. |

### 5.9.10. Class ModeSelectorPane extends BasePane

This class is the pane which player can choose mode for gameplay.

**Field**

| Name | Description |
|---|---|
| - final String defaultFont | Default font of the pane |
| - MainMenuScreen canvas | Canvas of the screen |
| - GridPane UIPane | Pane that consists of all labels and buttons. |

**Constructor**

| Name | Description |
|---|---|
| + ModeSelectorPane() | Initialize main menu canvas and add this canvas to this pane.<br>Draw canvas and initialize UI. |

Method

| Name | Description |
|---|---|
| + void draw() | Draw the background of main menu. |
| + void initUI() | Initialize UI of the UIPane which include header and 2 buttons for selecting mode. |
| + void setUIConstraint() | Set constraint of UI pane in row and column. |
| - void initSlotGrid() | Initialize buttons for selecting mode. Set on mouse clicked event handling:<br>- If story mode is clicked, play game audio, set gameplayMode in GameLogic to 1, and create level 1 of story mode.<br>- If puzzle mode is clicked, play game audio, set gameplayMode in GameLogic to 2, and initialize puzzle mode. |

### 5.9.11. Class SlotPane extends StackPane

This class is small slot pane in inventory pane, player can choose the slot to use items.

Field

| Name | Description |
|---|---|
| - int slotNum | Slot number |
| - Label amountLabel | Label that show amount of items left |

Constructor

| Name | Description |
|---|---|
| + SlotPane(int slot) | Set slotNum to slot number and initialize UI. |

**Method**

| Name | Description |
|---|---|
| + void initUI() | Initialize UI with image of items and label showing amount of items in the bottom right of the slot pane. |
| + void setAmountLabel(int num) | Set text of amount label to specific number. |

## 5.10. Package tile.base

This package contains base of tile in the game.

### 5.10.1. Interface Changable

This interface provides methods for tiles that is changable from any reason.

**Method**

| Name | Description |
|---|---|
| + *Tile getChangedTile()* | This method will be called when tile is going to change. This method will return changed tile from original one. |

### 5.10.2. Interface Fillable

This interface provides methods for tiles that allow crate to drop.

**Method**

| Name | Description |
|---|---|
| + *Tile getChangedTile()* | This method will be called when crate is dropped. This method will return changed tile from original one. |

### 5.10.3. Interface Fillable

This interface is marker interface for tiles that player and crate can be moved.

### 5.10.4. Abstract Class Tile

This abstract class is the base of all tiles.

**Field**

| Name | Description |
|---|---|
| - String name | Name of the tile |
| - Image img | Image of the tile |

**Constructor**

| Name | Description |
|---|---|
| + Tile(String imgName, String name) | Set image of this tile to be related with image name. Set name of the tile. |

**Method**

| Name | Description |
|---|---|
| + String getName() | Getter Method |
| + Image getImg() | |
| + void setName(String name) | Setter Method |
| + void setImg(Image img) | |

### 5.10.5. Abstract Class NormalTile extends Tile implements Walkable

This abstract class is the normal which player and crate can walk through.

**Constructor**

| Name | Description |
|---|---|
| + NormalTile(String imgName, String name) | Set image of this tile to be related with image name. Set name of the tile. |

### 5.10.6. Abstract Class WaterTile extends Tile implements Fillable

This abstract class is the water which crate can be filled.

**Constructor**

| Name | Description |
|---|---|
| + WaterTile(String imgName, String name) | Set image of this tile to be related with image name. Set name of the tile. |

**Method**

| Name | Description |
|---|---|
| + *Tile getChangedTile()* | This method will be called when crate is dropped. This method will return changed tile from original one. |

### 5.10.7. Abstract Class ColorPlatform extends Tile implements Walkable, Changable

This abstract class is color platform which can be inactivate by sensor entity.

**Constructor**

| Name | Description |
|---|---|
| + ColorPlatform(String imgName, String name) | Set image of this tile to be related with image name. Set name of the tile. |

**Method**

| Name | Description |
|---|---|
| + *Tile getChangedTile()* | This method will be called when sensor status is changed. This method will return changed tile from original one. |

### 5.10.8. Abstract Class ColorEmptyPlatform extends Tile implements Changable

This abstract class is color empty platform which can be activate by sensor entity.

**Constructor**

| Name | Description |
|---|---|
| + ColorEmptyPlatform(String imgName, String name) | Set image of this tile to be related with image name. Set name of the tile. |

**Method**

| Name | Description |
|---|---|
| + *Tile getChangedTile()* | This method will be called when sensor status is changed. This method will return changed tile from original one. |

## 5.11. Package tile.coloremptyplatform

This package contains every kind of color platform that is inactive.

### 5.11.1. Class RedEmptyPlatform extends ColorEmptyPlatform

This class represent red platform that is inactive. Player can active this platform by active red button sensor.

Constructor

| Name | Description |
|---|---|
| + RedEmptyPlatform() | Set image of this tile to red empty platform. Set name of the tile to “Red Empty Platform”. |

Method

| Name | Description |
|---|---|
| + Tile getChangedTile() | Return changed tile which is Red Platform |

### 5.11.2. Class OrangeEmptyPlatform extends ColorEmptyPlatform

This class represent orange platform that is inactive. Player can active this platform by active orange button sensor.

Constructor

| Name | Description |
|---|---|
| + OrangeEmptyPlatform() | Set image of this tile to orange empty platform. Set name of the tile to “Orange Empty Platform”. |

Method

| Name | Description |
|---|---|
| + Tile getChangedTile() | Return changed tile which is Orange Platform |

### 5.11.3. Class GreenEmptyPlatform extends ColorEmptyPlatform

This class represent green platform that is inactive. Player can active this platform by active green button sensor.

**Constructor**

| Name | Description |
|---|---|
| + GreenEmptyPlatform() | Set image of this tile to green empty platform. Set name of the tile to “Green Empty Platform”. |

**Method**

| Name | Description |
|---|---|
| + Tile getChangedTile() | Return changed tile which is Green Platform |

### 5.11.4. Class BlueEmptyPlatform extends ColorEmptyPlatform

This class represent blue platform that is inactive. Player can active this platform by active blue button sensor.

**Constructor**

| Name | Description |
|---|---|
| + BlueEmptyPlatform() | Set image of this tile to blue empty platform. Set name of the tile to “Blue Empty Platform”. |

**Method**

| Name | Description |
|---|---|
| + Tile getChangedTile() | Return changed tile which is Blue Platform |

### 5.11.5. Class PurpleEmptyPlatform extends ColorEmptyPlatform

This class represent purple platform that is inactive. Player can active this platform by active purple button sensor.

**Constructor**

| Name | Description |
| --- | --- |
| + PurpleEmptyPlatform() | Set image of this tile to purple empty platform. Set name of the tile to “Purple Empty Platform”. |

**Method**

| Name | Description |
| --- | --- |
| + Tile getChangedTile() | Return changed tile which is Purple Platform |

## 5.12. Package tile.colorplatform

This package contains every kind of color platform that is active.

### 5.12.1. Class RedPlatform extends ColorPlatform

This class represent red platform that is active. Player can inactive this platform by active red button sensor.

**Constructor**

| Name | Description |
| --- | --- |
| + RedPlatform() | Set image of this tile to red platform. Set name of the tile to “Red Platform”. |

**Method**

| Name | Description |
| --- | --- |
| + Tile getChangedTile() | Return changed tile which is Red Empty Platform |

### 5.12.2. Class OrangePlatform extends ColorPlatform

This class represent orange platform that is active. Player can inactive this platform by active orange button sensor.

**Constructor**

| Name | Description |
|---|---|
| + OrangePlatform() | Set image of this tile to orange platform. Set name of the tile to “Orange Platform”. |

**Method**

| Name | Description |
|---|---|
| + Tile getChangedTile() | Return changed tile which is Orange Empty Platform |

### 5.12.3. Class GreenPlatform extends ColorPlatform

This class represent green platform that is active. Player can inactive this platform by active green button sensor.

**Constructor**

| Name | Description |
|---|---|
| + GreenPlatform() | Set image of this tile to green platform. Set name of the tile to “Green Platform”. |

**Method**

| Name | Description |
|---|---|
| + Tile getChangedTile() | Return changed tile which is Green Empty Platform |

### 5.12.4. Class BluePlatform extends ColorPlatform

This class represent blue platform that is active. Player can inactive this platform by active blue button sensor.

Constructor

| Name | Description |
|---|---|
| + BluePlatform() | Set image of this tile to blue platform. Set name of the tile to “Blue Platform”. |

Method

| Name | Description |
|---|---|
| + Tile getChangedTile() | Return changed tile which is Blue Empty Platform |

### 5.12.5. Class PurplePlatform extends ColorPlatform

This class represent purple platform that is active. Player can inactive this platform by active purple button sensor.

Constructor

| Name | Description |
|---|---|
| + PurplePlatform() | Set image of this tile to purple platform. Set name of the tile to “Purple Platform”. |

Method

| Name | Description |
|---|---|
| + Tile getChangedTile() | Return changed tile which is Purple Empty Platform |

## 5.13. Package tile.normal

This package contains every normal tiles that player and crate can walk through.

### 5.13.1. Class CrateOnDeepStone extends NormalTile

This class represent normal “Crate on Deep Stone” tile.

**Constructor**

| Name | Description |
|---|---|
| + CrateOnDeepStone() | Set image of this tile to crate on deep stone. Set name of the tile to “Crate on Deep Stone”. |

### 5.13.2. Class CrateOnDirt extends NormalTile

This class represent normal “Crate on Dirt” tile.

**Constructor**

| Name | Description |
|---|---|
| + CrateOnDirt() | Set image of this tile to crate on dirt. Set name of the tile to “Crate on Dirt”. |

### 5.13.3. Class CrateOnSand extends NormalTile

This class represent normal “Crate on Sand” tile.

**Constructor**

| Name | Description |
|---|---|
| + CrateOnSand() | Set image of this tile to crate on sand. Set name of the tile to “Crate on Sand”. |

### 5.13.4. Class CrateOnStone extends NormalTile

This class represent normal “Crate on Stone” tile.

**Constructor**

| Name | Description |
|---|---|
| + CrateOnStone() | Set image of this tile to crate on stone. Set name of the tile to “Crate on Stone”. |

### 5.13.5. Class DeepStone extends NormalTile

This class represent normal “Deep Stone” tile.

**Constructor**

| Name | Description |
|---|---|
| + DeepStone() | Set image of this tile to deep stone. Set name of the tile to “Deep Stone”. |

### 5.13.6. Class GradientStone extends NormalTile

This class represent normal “Gradient Stone” tile.

**Constructor**

| Name | Description |
|---|---|
| + GradientStone() | Set image of this tile to gradient stone. Set name of the tile to “Gradient Stone”. |

### 5.13.7. Class Grass extends NormalTile

This class represent normal “Grass” tile.

**Constructor**

| Name | Description |
|---|---|
| + Grass() | Set image of this tile to grass. Set name of the tile to “Grass”. |

### 5.13.8. Class Ice extends NormalTile

This class represent normal “Ice” tile. Player and crate and slide on this tile.

**Constructor**

| Name | Description |
|---|---|
| + Ice() | Set image of this tile to ice. Set name of the tile to “Ice”. |

### 5.13.9. Class Sand extends NormalTile

This class represent normal “Sand” tile.

**Constructor**

| Name | Description |
|---|---|
| + Sand() | Set image of this tile to sand. Set name of the tile to “Sand”. |

### 5.13.10. Class Snow extends NormalTile

This class represent normal “Snow” tile.

**Constructor**

| Name | Description |
|---|---|
| + Snow() | Set image of this tile to snow. Set name of the tile to “Snow”. |

### 5.13.11. Class Stone extends NormalTile

This class represent normal “Stone” tile.

**Constructor**

| Name | Description |
|---|---|
| + Stone() | Set image of this tile to stone. Set name of the tile to “Stone”. |

### 5.13.12. Class WoodenPlatform extends NormalTile

This class represent normal “Wooden Platform” tile.

**Constructor**

| Name | Description |
|---|---|
| + WoodenPlatform() | Set image of this tile to wooden platform. Set name of the tile to “Wooden Platform”. |

## 5.14. Package tile.water

This package contains every water tiles that player cannot walk through but crate can be dropped in this tile.

### 5.14.1. Class WaterOnDeepStone extends WaterTile

This class represent normal “Water on Deep Stone” tile.

**Constructor**

| Name | Description |
|---|---|
| + WaterOnDeepStone() | Set image of this tile to water on deep stone. Set name of the tile to “Water on Deep Stone”. |

**Method**

| Name | Description |
|---|---|
| + Tile getChangedTile() | Return changed tile which is Crate on Deep Stone |

### 5.14.2. Class WaterOnDirt extends WaterTile

This class represent normal “Water on Dirt” tile.

Constructor

| Name | Description |
|---|---|
| + WaterOnDirt() | Set image of this tile to water on dirt. Set name of the tile to “Water on Dirt”. |

Method

| Name | Description |
|---|---|
| + Tile getChangedTile() | Return changed tile which is Crate on Dirt |

### 5.14.3. Class WaterOnSand extends WaterTile

This class represent normal “Water on Sand” tile.

Constructor

| Name | Description |
|---|---|
| + WaterOnSand() | Set image of this tile to water on sand. Set name of the tile to “Water on Sand”. |

Method

| Name | Description |
|---|---|
| + Tile getChangedTile() | Return changed tile which is Crate on Sand |

### 5.14.4. Class WaterOnStone extends WaterTile

This class represent normal “Water on Stone” tile.

**Constructor**

| Name | Description |
|---|---|
| + WaterOnStone() | Set image of this tile to water on stone. Set name of the tile to “Water on Stone”. |

**Method**

| Name | Description |
|---|---|
| + Tile getChangedTile() | Return changed tile which is Crate on Stone |