Yakiniku the last dinner

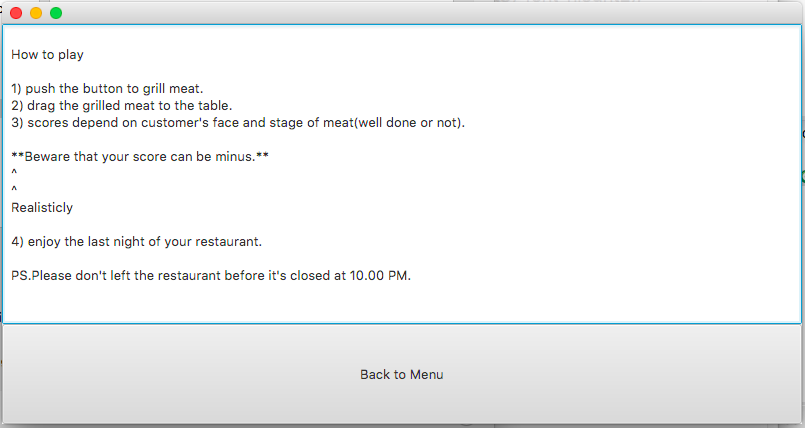
1.Overview

เกมนี้มีชื่อว่า Yakiniku THE LAST DINNER เป็นเกมที่เราจะได้สวมบทบาทเป็นเจ้าของร้านเนื้อย่างที่กำลังจะปิดทำการ



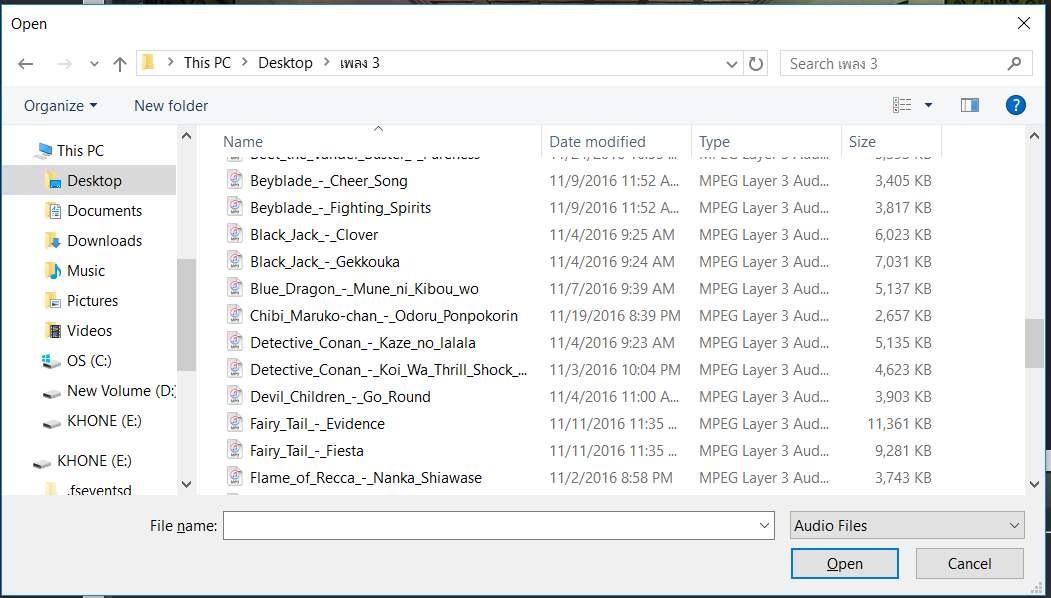
ภาพที่ 1 main screen เมื่อเข้าเกม

ใน main screen เราจะสามารถกด Enter เพื่อเริ่มเกม กดปุ่ม H เพื่ออ่านวิธีการเล่น



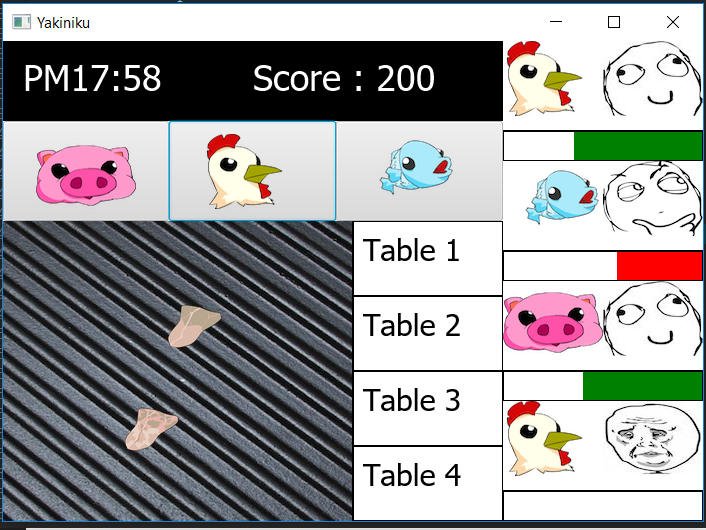
ภาพที่ 2 How to play

และสามารถกดปุ่ม B เพื่อเลือก background music จากในเครื่องได้ แต่เพลงที่เลือกนั้นจะต้องมีความยาวไม่ต่ำกว่า 20 วินาที



ภาพที่ 3 หน้าต่างเลือก BGM

ถ้า background music ไม่ถูกเลือกจะเป็นเพลง default ที่ถูกตั้งไว้แล้ว



ภาพที่ 4 game screen ขณะ ที่เล่นเกม

โดยในเกมเราจะต้องปิ้งเนื้อ และเสิร์ฟให้แต่ละโต๊ะ ก่อนที่ความอดทนของลูกค้าจะหมด และเนื้อจะต้องสุกกำลังดี เพื่อที่เราจะได้คะแนน ถ้าไม่เช่นนั้น คะแนนอาจจะติดลบได้ ร้านนี้เปิดทำการตั้งแต่ 16.00 น. จนถึง 22.00 น. เราสามารถกดปุ่ม Enter เพื่อ pause ระหว่างเล่นเกมได้



ภาพที่ 5 เมื่อเกมจบ

เมื่อร้านปิด เกมจะจบลง และหน้าต่าง game end จะขึ้นเพื่อแสดงคะแนนในรอบนั้นๆ จากนั้น เมื่อเราปิดหน้าต่าง game end เกมจะกลับสู่หน้า main screen

1.1 เนื้อ

* มี 3 ประเภท



1.หมู มี ระดับความสุกอยู่ 5 ขั้น ความสุกที่ดีที่สุดอยู่ที่ขั้นที่ 4



2.ไก่ มีระดับความสุกอยู่ 4 ขั้น ความสุกที่ดีที่สุดอยู่ที่ขั้นที่ 3



3.ปลา มีระดับความสุกอยู่ 3 ขั้น ความสุกที่ดีที่สุดอยู่ที่ขั้นที่ 2

1.2 กระทะ

เมื่อกดปุ่มที่อยู่ข้างบนกระทะ เนื้อจะถูกนำลงกระทะ เนื้อที่อยู่บนกระทะจะค่อยๆสุกไปตามเวลา แต่ต้องระวัง เพราะเนื้อสามารถไหม้ได้ แล้วเนื้อที่ไหม้ที่ถูกนำไปเสิร์ฟจะไม่ได้คะแนน หรือคะแนนอาจจะติดลบได้ นอกจากนั้น เราสามารถย่างเนื้อบนกระทะได้จำกัด เพราะฉะนั้นจะต้องวางแผนให้ดีในการย่างอีกด้วย



ภาพที่ 6 เนื้อที่ไหม้

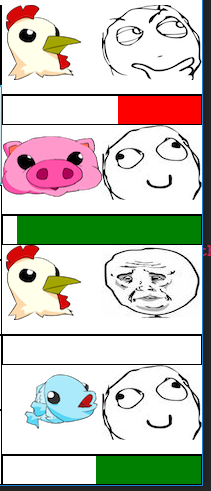
1.3 โต๊ะของลูกค้า

เมื่อเราทำการย่างเนื้อจนสุกได้ที่ เราจะต้องทำการลากเนื้อนั้นไปเสิร์ฟตามโต๊ะ



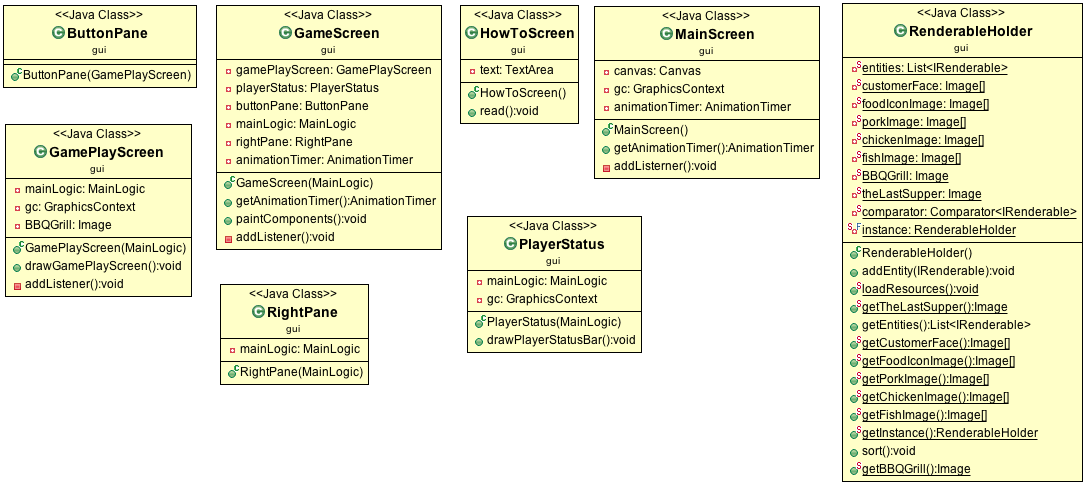
ภาพที่ 7 โต๊ะของลูกค้า

โดยเราต้องคอยสังเกต อาหารที่ลูกค้าสั่งและอารมณ์ของลูกค้าแต่ละโต๊ะว่าใกล้หมดความอดทนหรือยัง แต่ละโต๊ะย่อมมีความอดทนต่างกัน และแน่นอนว่าการเสิร์ฟอาหารให้ผิด ย่อมทำให้คะแนนไม่เพิ่ม

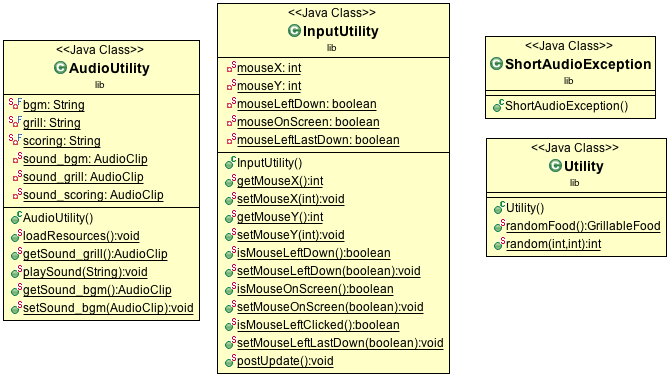


ภาพที่ 8 อาหารที่สั่งและอารมณ์ของลูกค้า

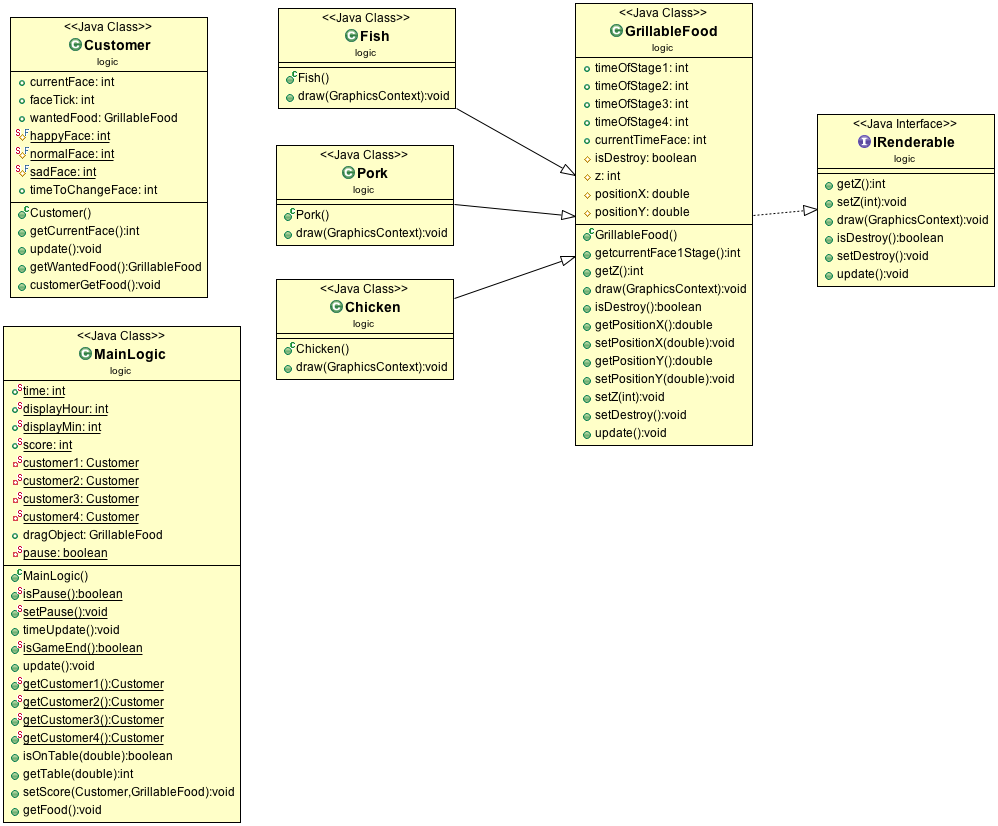
2.UML diagram



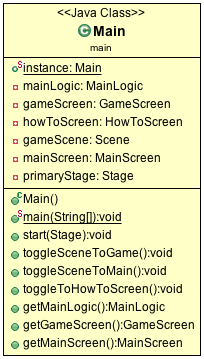
package gui UML diagram



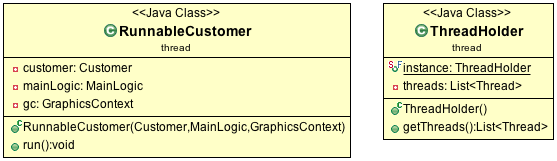
package lib UML diagram



package logic UML diagram



package main UML diagram



package thread UML diagram

3.Class details

3.1. Class “main.Main” extends Application

3.1.1. Fields

* + public static Main instance;
  + private MainLogic mainLogic;
  + private GameScreen gameScreen;
  + private HowToScreen howToScreen;
  + private Scene gameScene;
  + private MainScreen mainScreen;
  + private Stage primaryStage;

3.1.2. Methods

* + public static void main(String[] args); launch the application.
  + public void start(Stage primaryStage); throws Exception; set up stage scene and logic.
  + public void toggleSceneToGame(); to toggle to game screen.
  + public void toggleSceneToMain(); to toggle to main screen.
  + public void toggleToHowToScreen(); to toggle to how to play screen.
  + getters for mainLogic, gameScreen, mainScreen

3.2 Class “lib.AudioUtility”

3.2.1. Fields

* private static final String bgm = "bgm.mp3";
* private static final String grill = "sound\_grill.m4a";
* private static final String scoring = "scoring.mp3";
* private static AudioClip sound\_bgm;
* private static AudioClip sound\_grill;
* private static AudioClip sound\_scoring;

3.2.2. Methods

* public static void loadResources();
* public static void playSound(String Identifier);
* getter and setter for sound\_bgm();

3.3. Class “lib.InputUtility”

3.3.1. Fields

* private static int mouseX, mouseY;
* private static boolean mouseLeftDown, mouseOnScreen;
* private static boolean mouseLeftLastDown;
* private static ArrayList<KeyCode> keyPressed;
* private static ArrayList<KeyCode> keyTriggered;

3.3.2. Methods

* getters and setters for mouseX, mouseY, mouseLastLeftDown, keyPressed, keyTriggered;
* public static boolean isMouseOnScreen(),isMouseLeftClicked();
* public static void postUpdate();

3.4. Class “lib.Utility”

3.4.1. Methods

* public static GrillableFood randomFood(); to random food.
* public static int random(int start, int end); to random int.

3.5 Class “lib.ShortAudioException” extends Exception

Exception สำหรับ AudioClip ที่สั้นไป

3.5.1. Constructor

* public ShortAudioException();

3.6. Interface “logic.IRenderable”

3.6.1. Fields

* int getZ();
* boolean isDestroy();

3.6.2. Methods

* void setZ(int z);
* void draw(GraphicsContext gc);
* void setDestroy();
* void update();

3.7. Class “logic.GrillableFood” implements IRenderableHolder

3.7.1. Fields

* public int timeOfStage1;
* public int timeOfStage2;
* public int timeOfStage3;
* public int timeOfStage4;
* public int currentTimeFace;
* protected boolean isDestroy;
* protected int z = 2;
* protected double positionX;
* protected double positionY;

3.7.2. Constructor

* public GrillableFood(); set currentTimeFace, isDestroy, position x&y.

3.7.3. Methods

* public int getcurrentFace1Stage(); get a current face stage.
* public void draw(GraphicsContext gc); draw food.
* public boolean isDestroy();
* getters and setters for x,y,z;
* public void setDestroy(); set isDestroy to true.
* public void update(); update currentTimeFace if food is on the pan.

3.8. Class “logic.Chicken” extends GrillableFood

3.8.1. Constructor

* public Chicken(); set timeOfStage1 = 360;timeOfStage2 = 720;timeOfStage3 = 1080;

3.8.2. Methods

* public void draw(GraphicsContext gc);

3.9. Class “logic.Fish” extends GrillableFood

3.9.1. Constructor

* public Fish(); set timeOfStage1 = 200;timeOfStage2 = 400;

3.9.2. Methods

* public void draw(GraphicsContext gc);

3.10. Class “logic.Pork” extends GrillableFood

3.10.1. Constructor

* public Pork();set this.timeOfStage1 = 300;timeOfStage2 = 600;timeOfStage3 = 900;this.timeOfStage4 = 1200;

3.10.2. Methods

* public void draw(GraphicsContext gc);

3.11. Class “logic.Customer”

3.11.1. Fields

* public int currentFace;
* public int faceTick;
* public GrillableFood wantedFood;
* protected static final int happyFace = 2;
* protected static final int normalFace = 1;
* protected static final int sadFace = 0;
* public int timeToChangeFace;

3.11.2. Constructor

* public Customer(); set wantedFood and timeToChangeFace by random; faceTick = 0; currentFace = happy;

3.11.3. Methods

* getters for currentFace & wantedFood
* public void customerGetFood();customers get a food
* public synchronized void update(); update faceTick

3.12. Class “logic.MainLogic”

3.12.1. Fields

* public static int time;
* public static int displayHour,displayMin;
* public static int score;
* private static Customer customer1;
* private static Customer customer2;
* private static Customer customer3;
* private static Customer customer4;
* public GrillableFood dragObject;
* private static boolean pause;

3.12.2. Constructor

* public MainLogic(); set score,time = 0; set 4 customers;set pause = false

3.12.3. Methods

* getter and setter for pause
* getters for all customers
* public void timeUpdate();
* public synchronized static boolean isGameEnd(); check that game end at 22.00 PM
* public void update(); update is game not pause or end
* public boolean isOnTable(double x); check that food is on the table
* public int getTable(double y); where the table is
* public synchronized void setScore(Customer customer,GrillableFood food);scoring
* public void isGetFood();check that customer get a food when it is on a table

3.13. Class “gui.RenderableHolder”

3.13.1. Fields

* private static List<IRenderable> entities;
* private static Image[] customerFace;
* private static Image[] foodIconImage;
* private static Image[] porkImage;
* private static Image[] chickenImage;
* private static Image[] fishImage;
* private static Image BBQGrill;
* private static Image theLastSupper;
* private static Comparator<IRenderable> comparator;
* private static final RenderableHolder instance = new RenderableHolder();

3.13.2. Constructor

* public RenderableHolder();build Image arrays

3.13.3. Methods

* public synchronized void addEntity(IRenderable entity); add an image to arrays
* public static void loadResources(); load image resources
* getters for customerFace, foodIconImage, porkImage, chickenImage, fishImage, BBQGrill, theLastSupper and instance
* public void sort(); sort all entities in arrays by its Z

3.14. Class “gui.PlayerStatus” extend Canvas

แถบแสดงเวลา และคะแนนของผู้เล่น

3.14.1. Fields

* private MainLogic mainLogic;
* private GraphicsContext gc;

3.14.2. Constructor

* public PlayerStatus(MainLogic mainLogic);

3.14.3. Methods

* public void drawPlayerStatusBar(); วาดแถบ Player status

3.15. Class “gui.ButtonPane” extends GridPane

ปุ่มเลือกอาหารที่จะนำไปย่าง

3.15.1. Constructor

* public ButtonPane(GamePlayScreen gamePlayScreen);set 3 buttons(pork, fish, chicken)

3.16. Class “gui.RightPane” extends GridPane

แถบ customers 4 คน ที่แสดงด้วย threads

3.16.1. Fields

* private MainLogic mainLogic;

3.16.2. Constructor

* public RightPane(MainLogic mainLogic); set 4 threads of runnable customers

3.17. Class “gui.GamePlayScreen” extends Canvas

หน้าจอในส่วนที่เป็นกระทะสำหรับย่างเนื้อ และโต๊ะสำหรับเสิร์ฟอาหาร

3.17.1. Fields

* private MainLogic mainLogic;
* private GraphicsContext gc;
* private Image BBQGrill;

3.17.2. Constructor

* public GamePlayScreen(MainLogic mainLogic);

3.17.3. Methods

* public void drawGamePlayScreen(); วาดแต่ละส่วน (กระทะ และ โต๊ะ)
* private void addListener(); setOnAction ต่างๆ

3.18. Class “gui.MainScreen” extends StackPane

หน้าจอเมื่อเข้าเกม

3.18.1. Fields

* private Canvas canvas;
* private GraphicsContext gc;
* private AnimationTimer animationTimer;

3.18.2. Constructor

* public MainScreen();set main screen

3.18.3. Methods

* getter for AnimationTimer
* private void addListerner(); add key Event (Enter to start, H to How to play, B to select BGM)

3.19. Class “gui.HowToScreen” extends BorderPane

หน้าจอแสดงวิธีการเล่นเกม

3.19.1. Fields

* private TextArea text;

3.19.2. Constructor

* public HowToScreen(); set How to play screen

3.19.3. Method

* public void read(); display a text

3.20. Class “gui.GameScreen” extends GridPane

หน้าจอเมื่อกดเริ่มเล่นเกม

3.20.1. Fields

* private GamePlayScreen gamePlayScreen;
* private PlayerStatus playerStatus;
* private ButtonPane buttonPane;
* private MainLogic mainLogic;
* private RightPane rightPane;
* private AnimationTimer animationTimer;

3.20.2. Constructor

* public GameScreen(MainLogic mainLogic); set the GameScreen by combine game play screen, player status, ButtonPane and RightPane

3.20.3. Methods

* getters for AnimationTimer and GamePlayScreen
* public void paintComponents();draw screen
* private void addListener();add setOnEvents

3.21. Class “thread.RunnableCustomer” implements Runnable

สร้าง thread ของ customers

3.21.1. Fields

* private Customer customer;
* private MainLogic mainLogic;
* private GraphicsContext gc;

3.21.2. Constructor

* public RunnableCustomer(Customer customer , MainLogic mainLogic ,GraphicsContext gc);

3.21.3. Methods

* public void run(); start thread by random time
* getter for customer

3.22. Class”thread.ThreadHolder”

ที่รวม thread

3.22.1. Field

* private List<Thread> threads; array that hold all threads

3.22.2. Method

* getter for threads