



Course Code : SWE305

Course Name : Game Design & Development

Lecturer : Valarmathie Gopalan

Academic Session : 2021/09

Assessment Title : Final Project - Apply Game Design Skills to Create a 2D Game

Submission Due Date : Week 16 (7 Jan 2022)

Prepared by : 

Student ID	Student Name
SWE1909766	Wong Ze Min

Date Received : \_\_\_\_\_

Feedback from the Lecturer:

Mark:

### **Own Work Declaration**

I/We hereby understand my/our work would be checked for plagiarism or other misconduct, and the softcopy would be saved for future comparison(s).

I/We hereby confirm that all the references or sources of citations have been correctly listed or presented and I/we clearly understand the serious consequence caused by any intentional or unintentional misconduct.

This work is not made on any work of other students (past or present), and it has not been submitted to any other courses or institutions before.

Signature:

A handwritten signature in black ink, appearing to read 'Wong Ze Min', with a long horizontal stroke extending to the right.

(WONG ZE MIN)

Date: 06/01/2022

## Contents

<b>Discussion .....</b>	<b>4</b>
<b>Game Theme/Background .....</b>	<b>4</b>
<b>Game Objectives.....</b>	<b>4</b>
<b>Target Users .....</b>	<b>4</b>
<b>Design .....</b>	<b>6</b>
<b>Implementation .....</b>	<b>9</b>
<b>Game Engine .....</b>	<b>9</b>
<b>Others .....</b>	<b>12</b>
<b>Basic/Main interactable game objects .....</b>	<b>13</b>
<b>Problems and Challenges.....</b>	<b>32</b>
<b>The Screen Layout .....</b>	<b>34</b>
<b>References.....</b>	<b>39</b>
<b>Sprites .....</b>	<b>39</b>
<b>Music.....</b>	<b>40</b>
<b>Scripts .....</b>	<b>42</b>
<b>Project Product Demonstration- YouTube video presentation link- .....</b>	<b>45</b>

## Discussion

### Game Theme/Background

The game Dummy Robot- Supper Take-Out Delivery is an integrated memory- *objects/items memorizing-* and puzzle- *maze solving-* game, packaged with a theme of take-out delivery in a futuristic neo cyberpunk city background. In which, the game involves elements including grabbing take-outs from restaurants upon the customer's order, exploring/shuttling around the city's neighbourhood at late night, probing on every side and corner of the neighbourhood's alley, and ultimately delivering take-outs to the hands of the customer.

### Game Objectives

The objective of the game is simple, which as the name take-out delivery suggests, namely to deliver take-outs to the customer's place. However, there are three main game challenges- *gaming facets-*, namely:

- Objects/Items memorizing

A set of take-out items ordered by the customers to be remembered. In which the take-out items will be shown only once. Upon the memorized items, respective items that correspond to the order shall be collected.

- Maze solving

A maze to be solved. In which the end point/exit of the maze manifested as the delivery location- *customer's location-* to be reached. In the given maze/map, exploring through the city's neighbourhood until reaching and delivering the take-outs to the customer's hands- *exit-*.

- Area/Space probing- *Game control constraint-*

The setting of the protagonist Dummy Robot is a robot with a defective vision and special sensors- *malfunctioning system embedded camera and radars-* yet it still can rely its collision sensors to probe and move around. In the game, upon the provided movement mechanism and settings, probing on every side and corner of the neighbourhood's alley, and ultimately delivering take-outs to the customer's location.

### Target Users

The game Dummy Robot- Supper Take-Out Delivery is designed to be a fun/friendly casual mini memory and puzzle game. The anticipated target players/consumers of the game are as follows:

- Children above age of 5 y/o  
Memory games and puzzle games are children friendly and good for their Intelligence quotient (IQ) development. Memory games and puzzle games are good for people of all ages.
- Casual gamers who wish to have quick gaming for relaxing, time-killing, etc.  
Mini puzzle games can be very effective for short entertainment, for example the game 2048 is one of the most successful casual games since its release in 2014. Which it is popular among people such as subway commuters, students, office working, etc. that they play 2048 whenever they are having a short break.

## Design

The idea of the game was inspired by an individual game project from Fabian Ros call Let's Jump- *\*Link: <https://code-projects.org/lets-jump-game-in-unity-engine-with-source-code/> -*. The game let's jump is a mobile game, which like the game Flappy Bird, it is an addictive tapping game with a gameplay shuttling across gaps to collect golds- *items*- and reach to the queen- *end/goal point*-. Where, through tapping the character moves and jumps diagonally, and whenever it hits/collides with surfaces/obstacles/borders, it will reflect to another direction. The demonstration of the game Let's Jump is as follow:

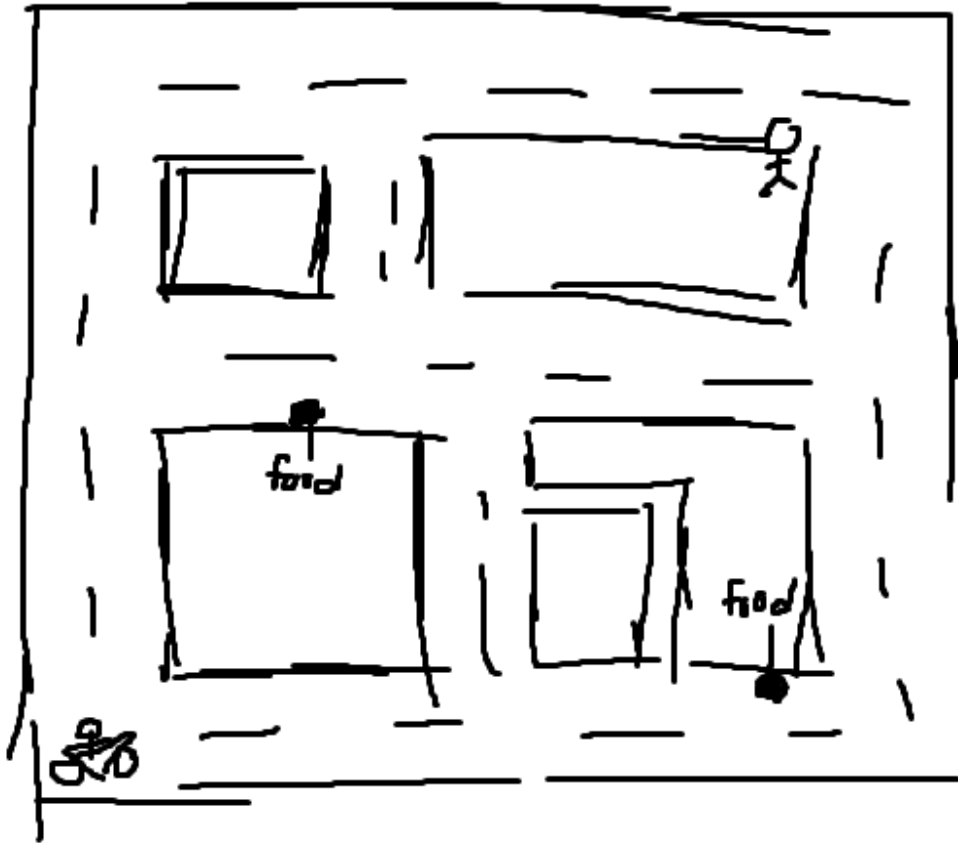


*\*Link: <https://code-projects.org/wp-content/uploads/2018/07/letsjump-gif.gif>*

The concept of shuttling across gaps to collect gold and reaching to the queen got me relate to the take-out delivery- *namely Grab Food*- instance in real life, where the driver goes to a place for grabbing a take-out and deliver to the customer's place. Hence the basic idea- *theme*- of the game, namely take-out delivery, where the concept- *of Let's Jump*- is being manifested as shuttling across city's neighbourhood and alley, collecting take-outs, and delivering to the customer's location.



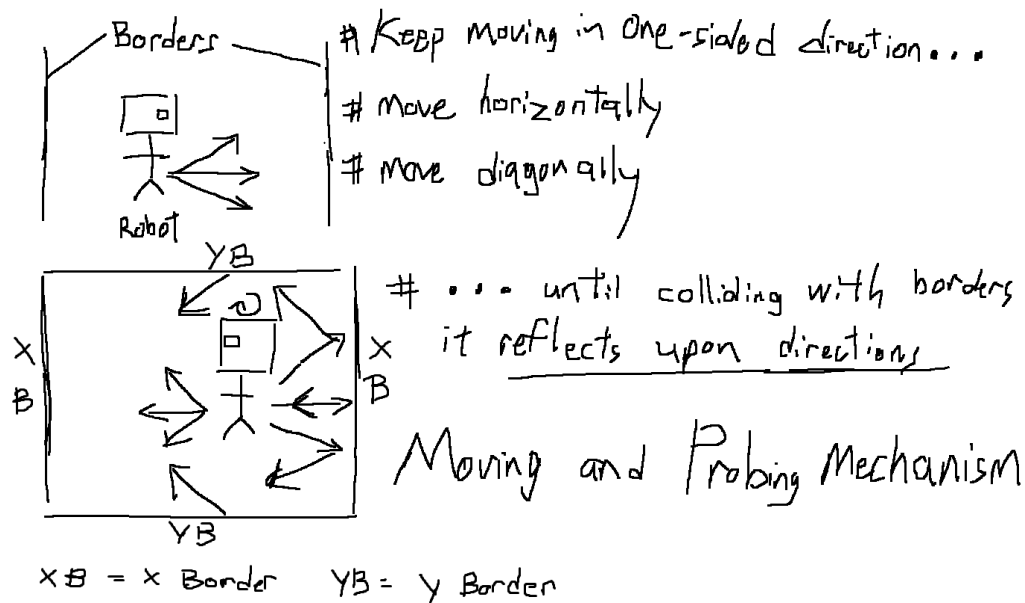
*(\*Resources- <https://www.techarp.com/business/grabfood-restaurant-availability/>)*



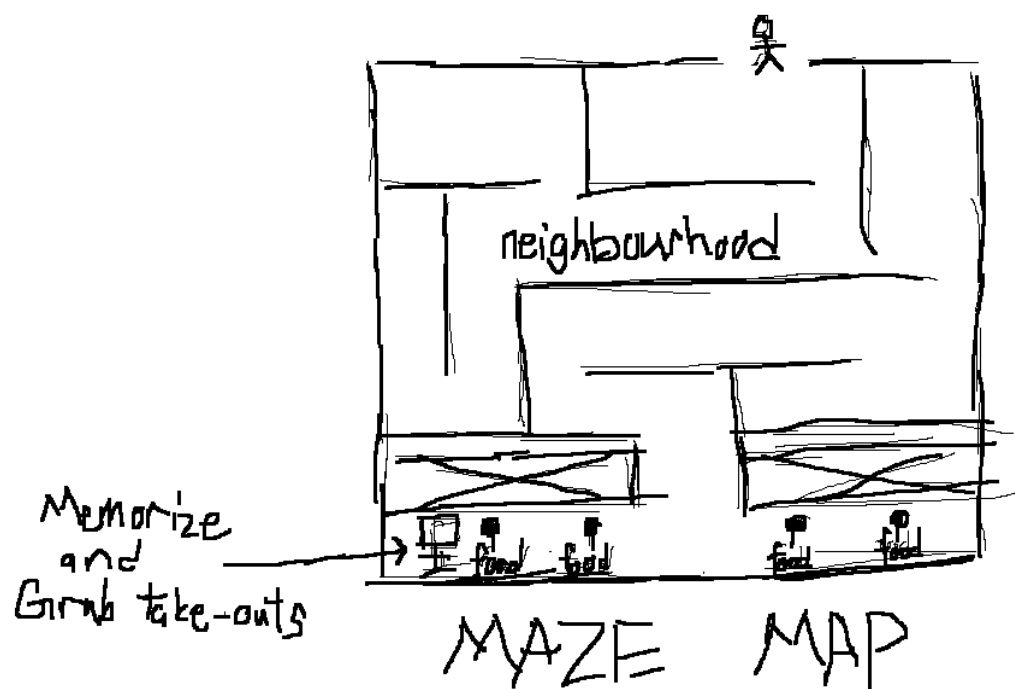
*#Game basic idea sketch*

Like the take-out delivery in real life, it was originally to make a motorcyclist as the playable character. However, a motorcyclist moving recklessly doesn't make any sense, which it is far-fetched that the motorcyclist can only move in one-sided directions- *horizontally and diagonally*- and reflect to another side only when making a collision. Therefore, the setting for having a character with defective vision, yet this setting was quickly abandoned given riding without ability to see is dangerous- *also it doesn't make any sense*-.

After several analysis, designing, and consideration hence the born of the protagonist, namely Dummy Robot, with a setting of defective vision and special sensors- *malfunctioning system embedded camera and radars*- where it relies its collision sensors to probe and move around. To further meets the setting a futuristic background, namely neo cyberpunk, is adopted and adapted into the gameplay. Thereupon, adding with gaming attribute, the final manifested integrated memory- *take-outs memorizing*- and maze puzzle- *destination map solving*- game.



## #Game mechanism design sketch



*# Final manifested integrated memory- take-outs memorizing- and maze puzzle-  
destination map solving- game sketch*



## Implementation

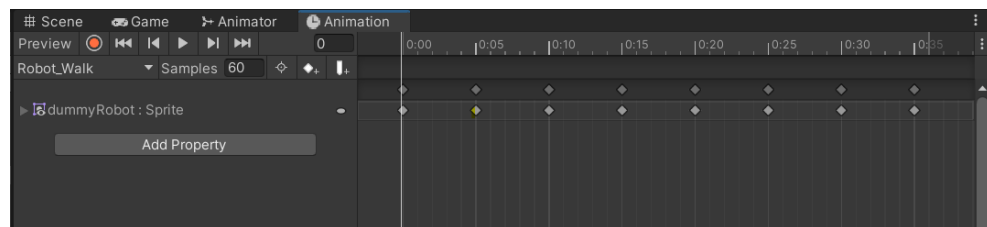
Summary of what objects you have created, what software's you have used, and what features in each software you have applied into your game. This section also can include screenshots and explanation about each item in your game such as objects, scripts.

### Game Engine

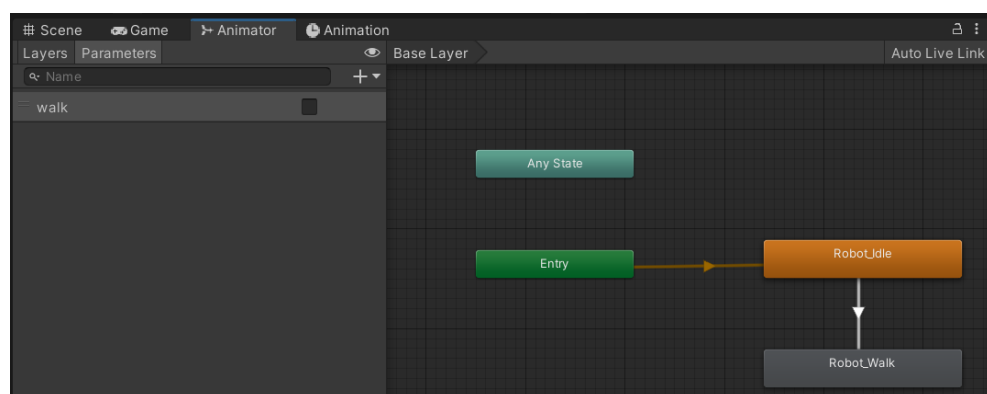
The adopted game engine for the development and implementation of the 2D game project is Unity- *Unity 2020.3.19f1 (64-bit)*-. In which the applied/utilized Unity technologies to realize the implementation include as follows:

- Unity technologies (Internal)
  - 2D Animation

Used for skeletal animation using Sprites.



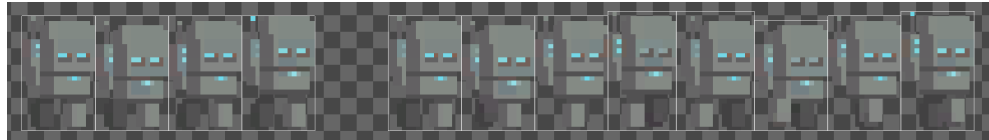
*#Used to set up the animation of the Dummy Robot, including idle animation and walking animation*



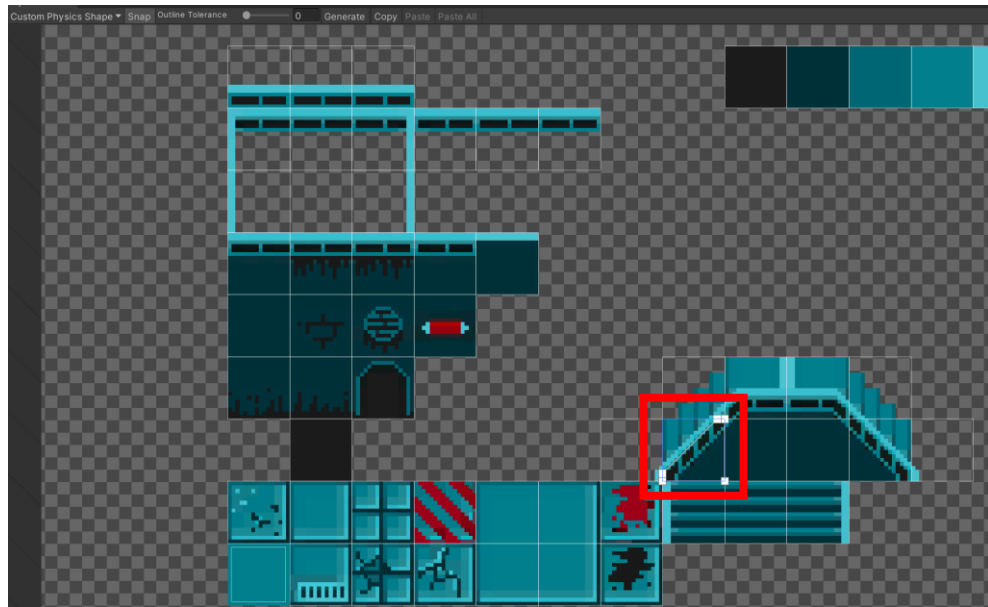
*#Used to set up and animate the state of animation of the Dummy Robot*

- 2D Sprite

Used to create and edit Sprite asset properties like pivot, borders and Physics shape.



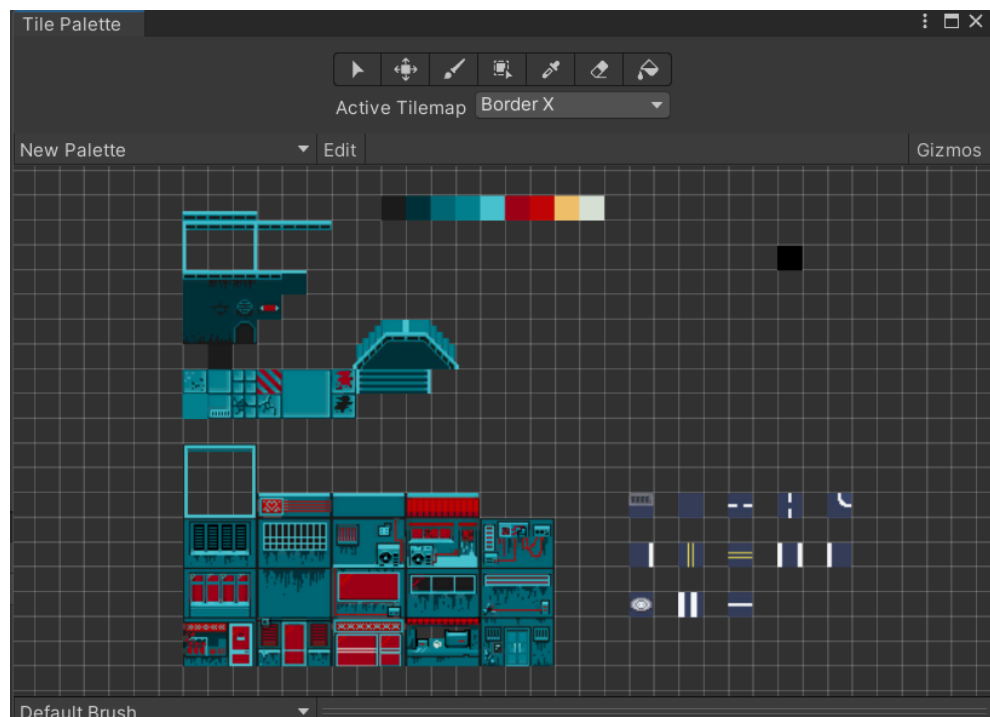
*#Used to create the sprite of the Dummy Robot*



*#Used to create the custom physics shape of the borders needed to build the map*

- 2D Tile-map Editor

Used for editing Tile-maps.



*#Used to set-up the tiles needed to build the map*

- Unity UI

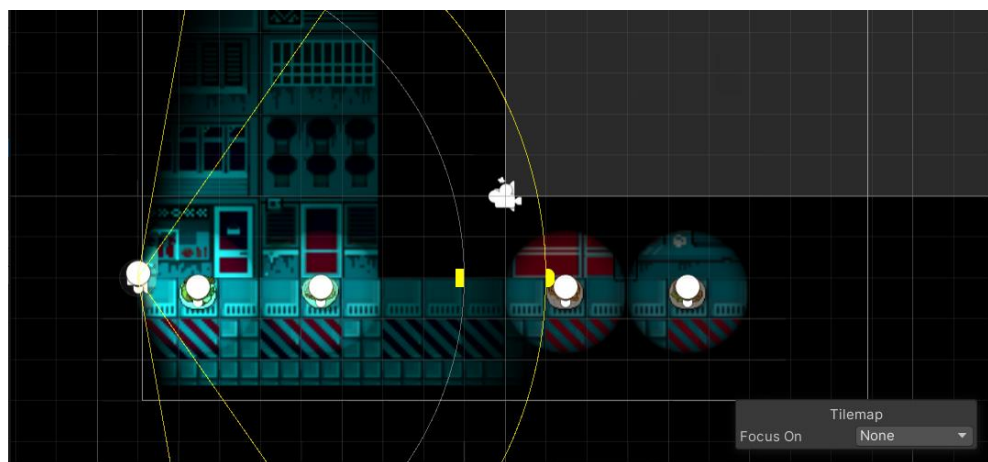
Used for developing User Interfaces (UI) for the game.



*#Used to set-up and implement the UI of the game*

- Universal Render Pipeline (URP)

Used to create optimized graphics- and specifically for the access to lighting physics features-



*#Used to set-up and implement the lighting of the game. Given the game concept supper take-out delivery, which lighting is critical to implement a late night environment and vibe.*

- Externals

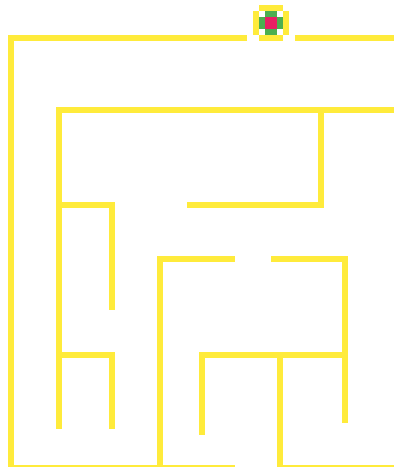
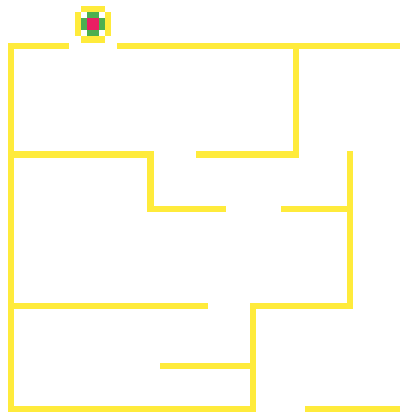
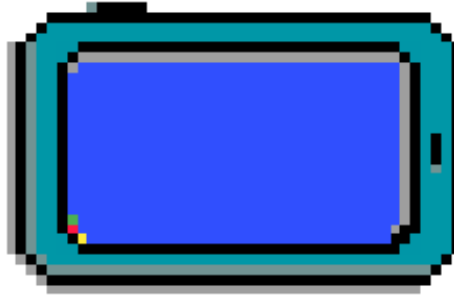
- Visual Studio Editor

Used as the main Integrated Development Environment (IDE) and the code editor for unity.

## Others

- PIXILART (\*Link: <https://www.pixilart.com/draw#>)

Used to create 2D assets/sprites



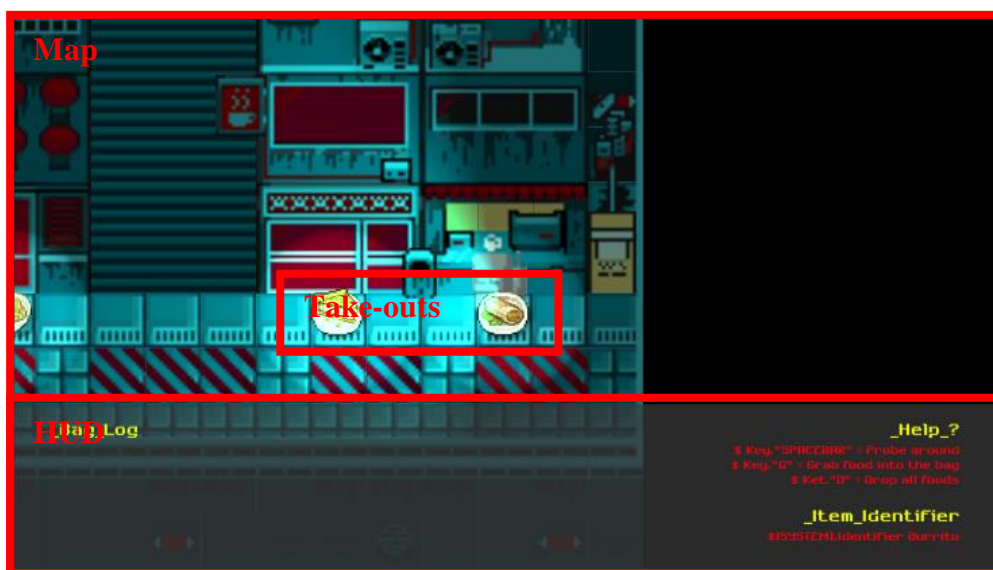
*#Used to create 2D assets/sprites*

## Basic/Main interactable game objects

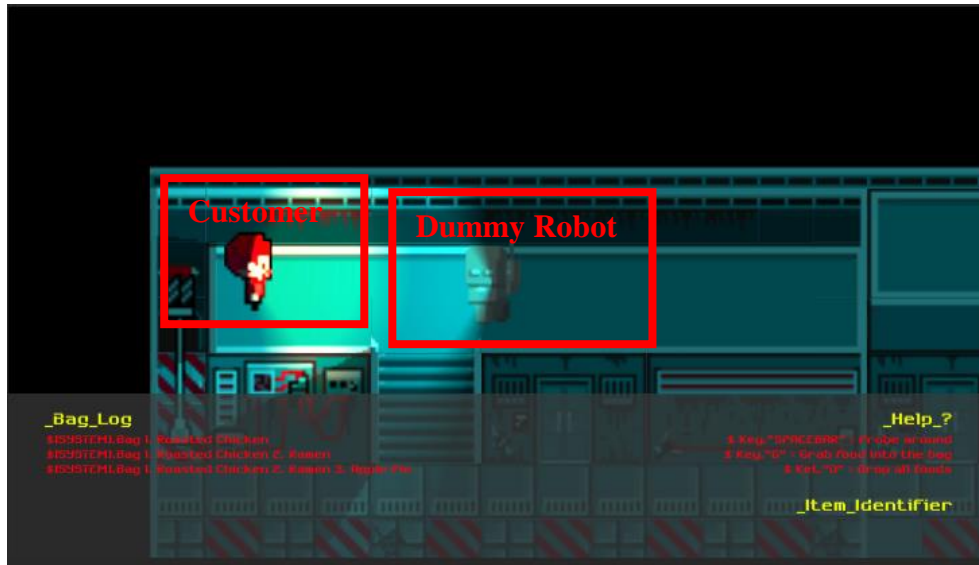
There are four main manifested categories of interactable game objects which include **characters**, **map**, **collectable items**, and **UI**. The overview of the interactable game objects are as follows:



*#Interactable game object- UI: Menu*



*#Interactable game object- collectable item: take-outs/foods, UI: Head-Up Display HUD, and manifested map*



*#Interactable game object- Playable character: Dummy Robot and NPC: customer*

## Characters

### **Playable character**

The main character controlled by the player to play the game. The featured interaction are as follows:

- Key.Spacebar- Toggle the key 'spacebar' to control the character, in which as aforementioned to probe on every side and corner of the environment- *map*- and move around the maze until reaching the customer's location.
- Key.G- Press the key 'g' to grab the take-outs/foods into the bag.
- Key.D- Press the key 'd' to drop all the grabbed take-out/foods from the bag.

### **Non-playable characters (NPCs)**

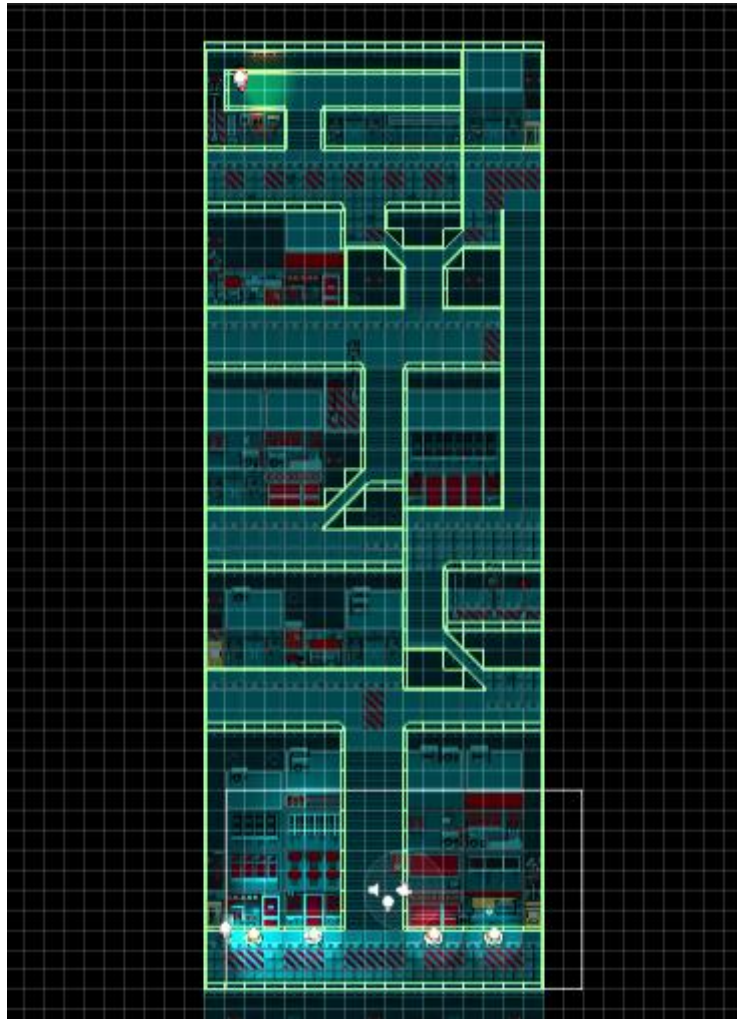
The customer who the player needs to serve, which to deliver the take-outs to its hands/location.



***#Interactable game object- Playable character: Dummy Robot and NPC: customer***

Map

The manifested maze/level of the game for the player to play. The map is being implemented with carefully placed borders- *highlighted in green*- that the player will be utilizing in the controlling of the character to reflect and move from directions to directions.



***#Interactable game object- Map: Manifested maze and collision borders***

Collectable items

The collectable items, namely the take-outs/foods, that the player can/needed to collect and grab and deliver to the hands of the ordering customer.





*#Interactable game object- Collectable items: Take-outs/Foods*

### User Interface (UI)

#### **Menu(s)**

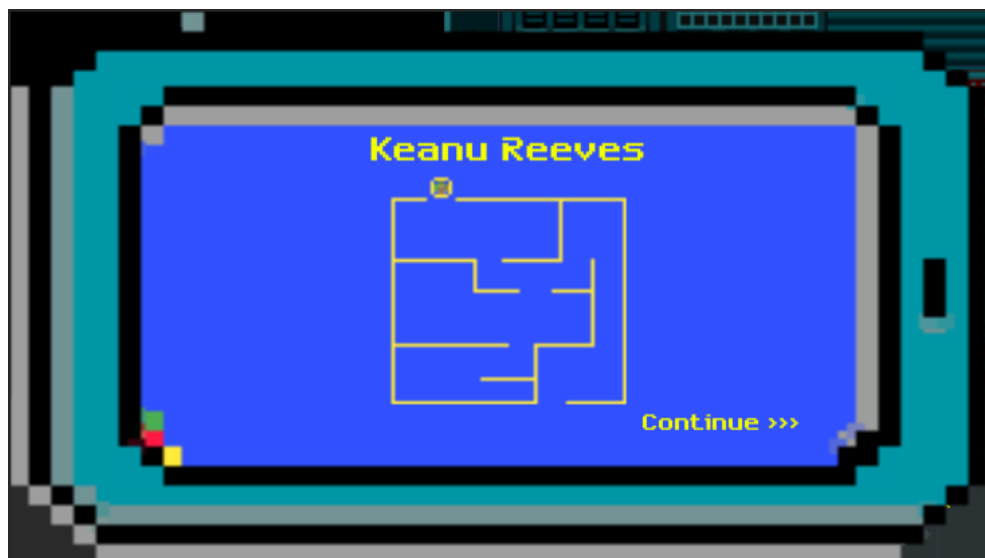
The user interfaces where the player used to interact with the game. The type interactions include as follows:

- Informative- Of interaction such as displaying and informing gameplay facets such as the scores, game victory- *wins*-, and game over- *loses*-.
- Controlling- Of interaction such as providing and allowing game control functions such as, play game, quit game, restart level, forward paragraph sections, etc.



*#Interactable game object- Menu (Controlling): Main menu*





*#Interactable game object- Menu (Informative and controlling): Take-out order interface*





*#Interactable game object- Menu (Informative and controlling): Win and lose menu(s) with scores report*



*#Interactable game object- Menu (Informative and controlling): Scoreboard*

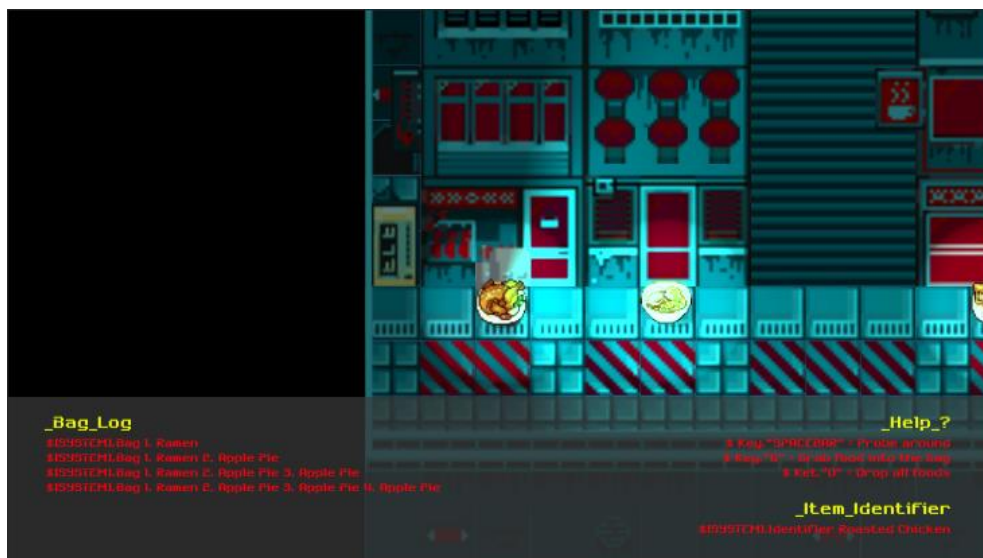


*#Interactable game object- Menu (Informative and controlling): Ending and credit menu*

## Head-Up Display (HUD)

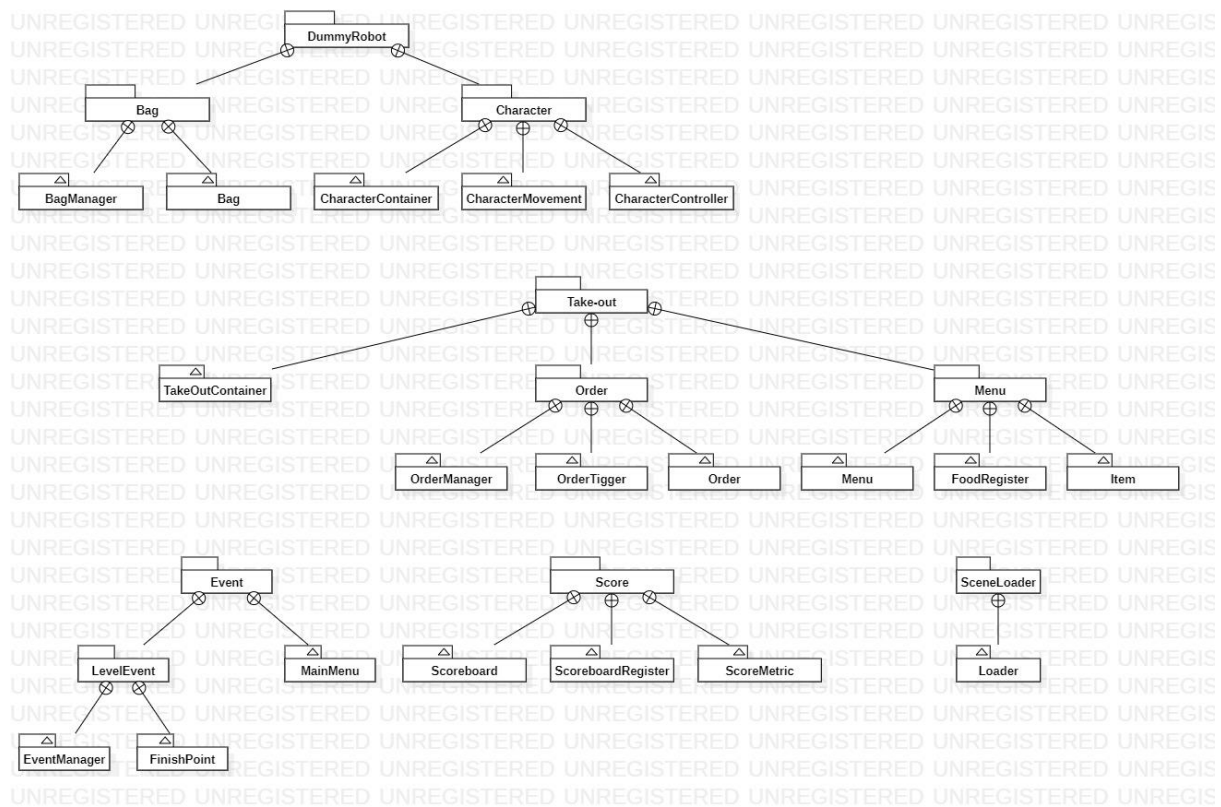
The user interfaces where the player used to interact with the state of the game. The type interactions include as follows:

- Informative- Of interaction such as displaying and informing gameplay facets such as the status of the bag inventory- *bag log*-, identifying the collectable items- *item identifier*-, and providing help and tutorial of the available controls- *help*-.

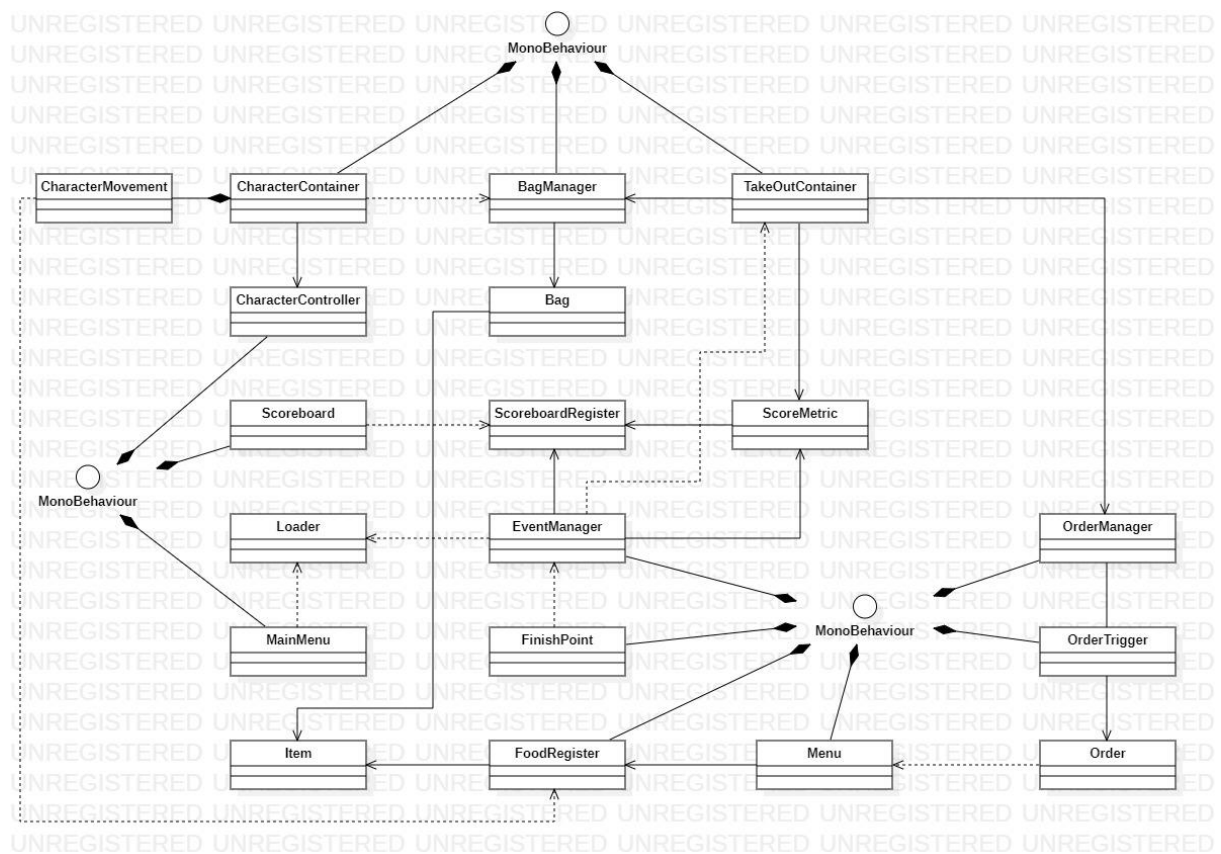


*#Interactable game object- Menu (Informative and controlling): HUD*

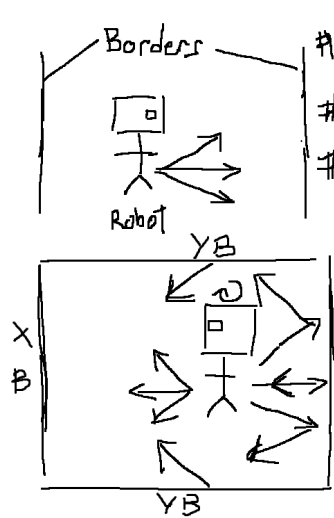
## Domain implementation objects




*#Overview of the domain scripts used to implement the domain logic of the game*




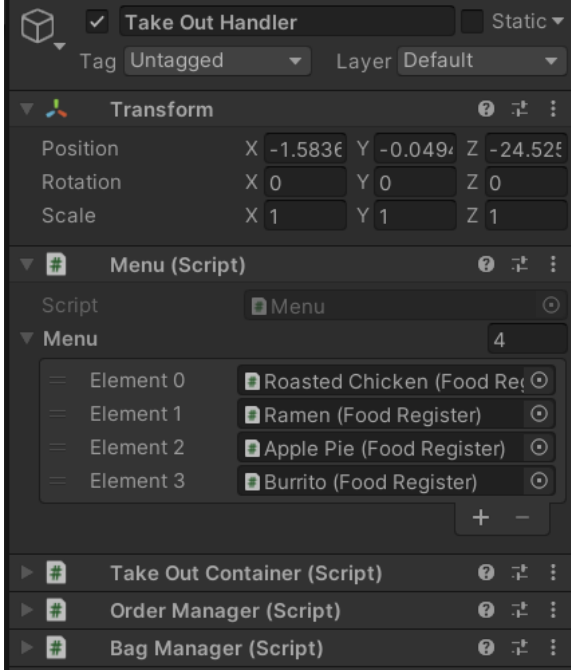
***#Overview of the domain model manifesting the domain logic and implementation of the game***

CharacterContainer	
CharacterMovement	
Description	<p>Handle player inputs and define the movement- <i>moving angle-</i> and movement algorithm- <i>border collision reflection-</i> of the character- <i>namely dummy robot-</i>.</p> <div style="display: flex; align-items: flex-start;"> <div style="flex: 1;">  <p style="text-align: center;">XB = x Border    YB = y Border</p> </div> <div style="flex: 1; padding-left: 20px;"> <p># Keep moving in one-sided direction...</p> <p># move horizontally</p> <p># move diagonally</p>   <p># ... until colliding with borders it reflects upon directions</p> <p><b>Moving and Probing Mechanism</b></p> </div> </div> <p align="center"><b><i>#The outcome of the implementation for the gameplay is as this requirement anticipated</i></b></p>
CharacterContainer	
Description	<p>Realize the handle and access of persistence abstract layer instances, including the transform.position, transform.rotation, and transform.scale- <i>namely the movements of the dummy robot and the attached point light-</i>. The aspects of access include as follows:</p> <ul style="list-style-type: none"> <li>• transform.position- Update the position of the sprite</li> <li>• transform.rotation - Update and tilt the angle of the sprite and point light- <i>to make it more lively-</i></li> <li>• transform.scale- Update to flip the sprite.</li> </ul> <p><u>Move</u></p>



	 <p><u>Tilt</u></p>   <p><u>Flip</u></p> 
--	---

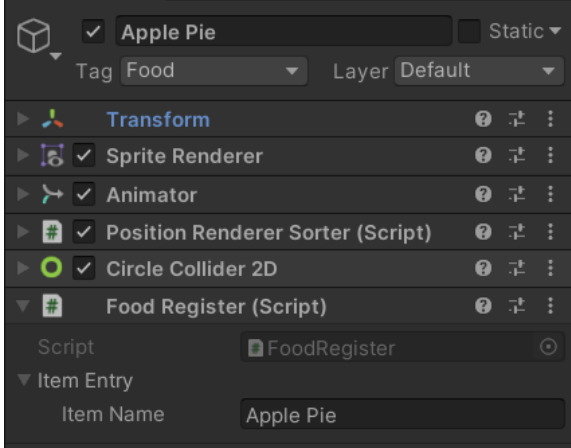
BagManager	
Description	<p>Realize the handle and access of persistence abstract layer instances/data, namely the Bag, including add food item, drop food item, and retrieve food items- <i>used by TakeOutContainer for order validation-</i>.</p>  <p><i>#An example of grabbing food items</i></p>


	 <p><i>#An example of dropping all grabbed food items</i></p>
Bag	
Description	Define persistence instances/data- <i>food item inside the bag-</i>

Menu	
Description	<p>Provide functionalities, including initialize food register and food item-  <i>assigning ID- and generate order- generate random food item-</i></p>  <p><i># Food register and food item initialization</i></p>



	  <p><i>#Generate order/food items</i></p>
FoodRegister	
Description	Realize the handle and access of persistence abstract layer instances/data, namely the Bag, including register food item entry, and get food item entry.



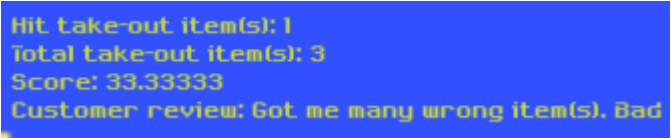
		 <p><i>#Food register entry</i></p>	
Item			
Description	Define persistence instances/data- <i>food item</i> -		


OrderManager	
Description	<p>Handle the incoming order requests- <i>fired from order trigger</i>- and display and present to the screen.</p>  <p><i>#An example of an order request being displayed and presented</i></p>
OrderTrigger	
Description	Implement the order set up- <i>declaring the customer's name, customer's location- map/map image</i> -, number food items- <i>order</i> - to generate, and the instant firing of order request.

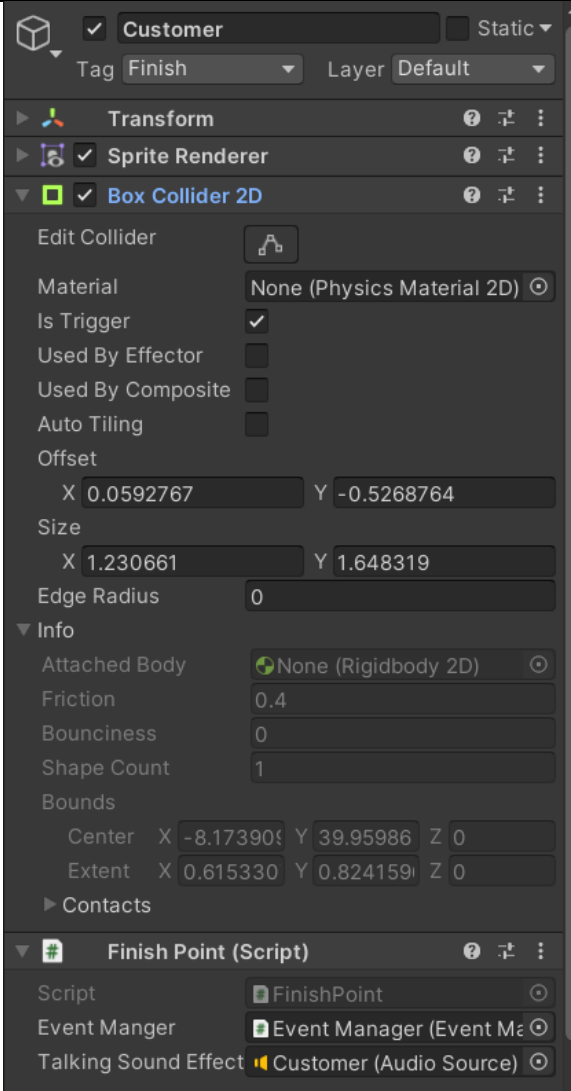
	 <p><i><b>#Order set up</b></i></p>
Order	
Description	Define persistence instances/data- <i>order/a set of take-out items ordered by the customers-</i>

TakeOutContainer	
Description	A “hub/interface” that realize and handle the data interaction of BagManger and OrderManger, including validate delivered take-out orders.

Scoreboard	
Description	Retrieve/access the score reports/results stored in the static class ScoreboardRegister and handle the retrived score reports/results of all levels and display and present to the screen.

	  <p><i>#An example of score results of all levels being displayed and presented</i></p>
ScoreMetric	
Description	<p>Define score metrics and handle the calculation of scores</p>  <p><i>#The score metric</i></p>
ScoreboardRegister	
Description	<p>Define <b>*static</b> persistence instances/data- <i>score result/report</i>-, that will be used across abstract spaces- <i>across from scene to scene</i>-</p>

EventManager	
Description	<p>Handle the incoming interrupt/level closing requests- <i>fired from FinishPoint</i>- and implement the closing tasks, including invoke delivered take-out orders validation function of TakeOutContainer, retrieve score report from ScoreMetric and display and present the result to the screen, and invoke load next level function from Loader.</p>  <p><i>#An example of level being closed with the score result displayed and presented. By clicking the button “Next level”, the subsequent level will be loaded.</i></p>
FinishPoint	
Description	<p>Realize the finish point set up- <i>calling the event manager</i>- which implement an end point trigger that handle the instant firing of interrupt/level closing request to Event Manager.</p>

	 <p><i>#Finish point set up</i></p>	
--	--	--

Loader	
Description	Define <b>*static</b> persistence instances/data- <i>scenes</i> -, that will be used across abstract spaces- <i>across from scene to scene</i> -, which handle and realize the transition and branching of scenes, namely the restarting of a level, transitioning from a level to the next level, and branching to the main menu and etc.

MainMenu	
Description	Handle player inputs, including play game and exit game- <i>where realized through invoking Loader functions</i> -



## Problems and Challenges

As aforementioned and emphasized all over the previous section, the core gaming elements and factors of the game Dummy Robot- Supper Take-Out Delivery is the movement mechanism. In which, the movement algorithm and the collision borders are the main problems and challenges where mainly manifested in the development and implementation of the game. The problems and challenging facets are as follows:

- Complex movement algorithm needed to be designed

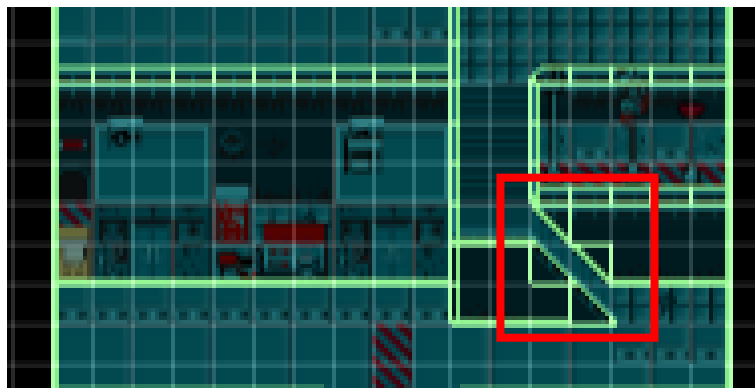
A considerable number of variables- *namely the aforementioned variables, including move, tilt angle, and flip*- needed to be designed and manifested precisely into the movement algorithm to implement and realize the movement mechanism.

- Fine implementation of borders is critical

It is essential for the borders to be finely set and configured so that the incorporation between the movement algorithm is effective and valid for realizing the movement mechanism.

The confrontation and containment to the problems and challenges were rather by “brute force”, namely to adjust the implementation of the algorithm and the physics shape configuration and position of borders “amateurly” and progressively until a satisfied and anticipated outcome is achieved.

Beside the basic movement mechanism- *the movement of the character*- problems, there is one noticeable incorporation problem between the movement algorithm and the borders’ physics configuration that is found to be tricky during the implementation. With an ineffective implementation of the incorporation between the movement algorithm and border’s physics shape, there will be a congestion problem when moving up the stairs.



***#Manifested congestion problem when Dummy Robot trying to move up the stairs***



The manifested congestion problem when Dummy Robot trying to move up the stairs is due to the ineffective implementation of the incorporation between the movement algorithm and border's physics shape. In which given that the and the gap with the positioning of borders and its physics shape is narrow- *where the character will stuck at the stair when tilting the movement-* and there was no such account taken into the implementation of the movement algorithm- *where the movement will not reflect when the character is moving horizontally thus resulting in a congested movement when moving up the stair-*.

The confrontation and containment to the problems and challenges were also rather by “brute force”, which progressively fixing until a satisfied and anticipated solution is found. In this case a trick is used, which by “knocking back” and flicker the position of the character with a small Y variance point of 0.1f whenever colliding with the y borders.

```
public void KnockBack(Vector2 pos)
{
    if (vehicleRigidBody.position.y < pos.y + 0.63)
    {
        transform.position = transform.position + new Vector3(0f, -0.1f, 0f);
    }
    else
    {
        transform.position = transform.position + new Vector3(0f, +0.1f, 0f);
    }
}
```

*#Solution to the congestion problem*

## The Screen Layout



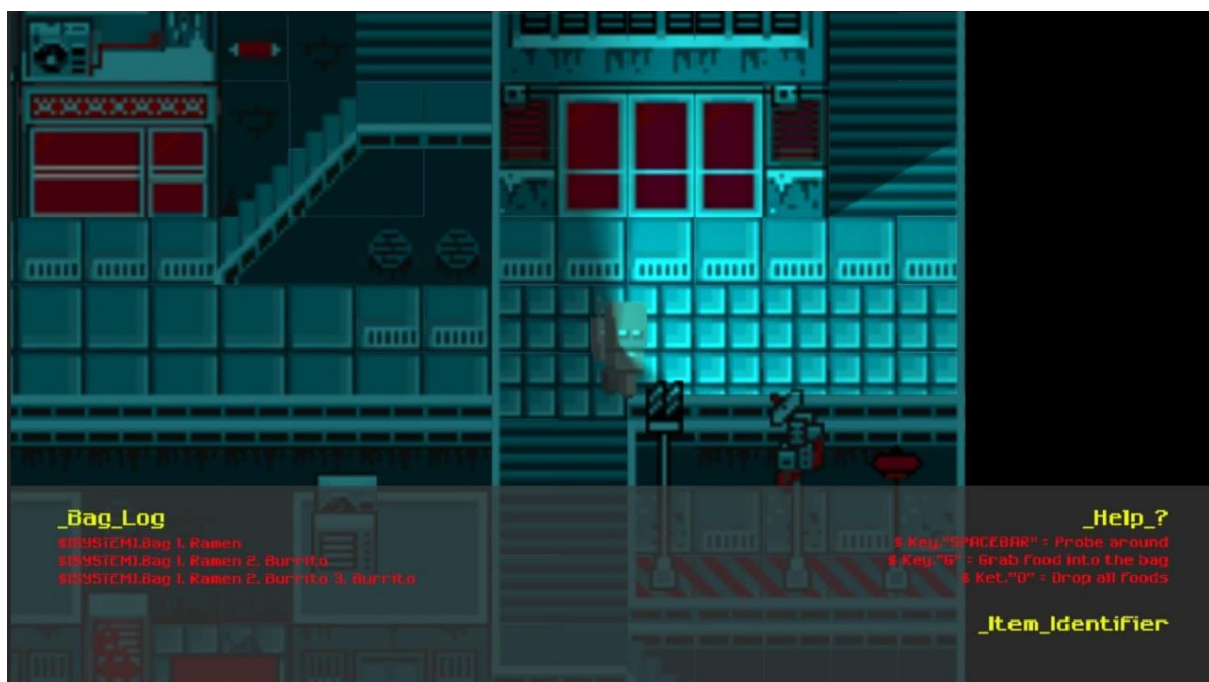
*#Main menu*



*#Take-out order interface (1): Take-out items ordered by the customers*



*#Take-out order interface (2): Maze map/Customer location*



*#Gameplay level 1*



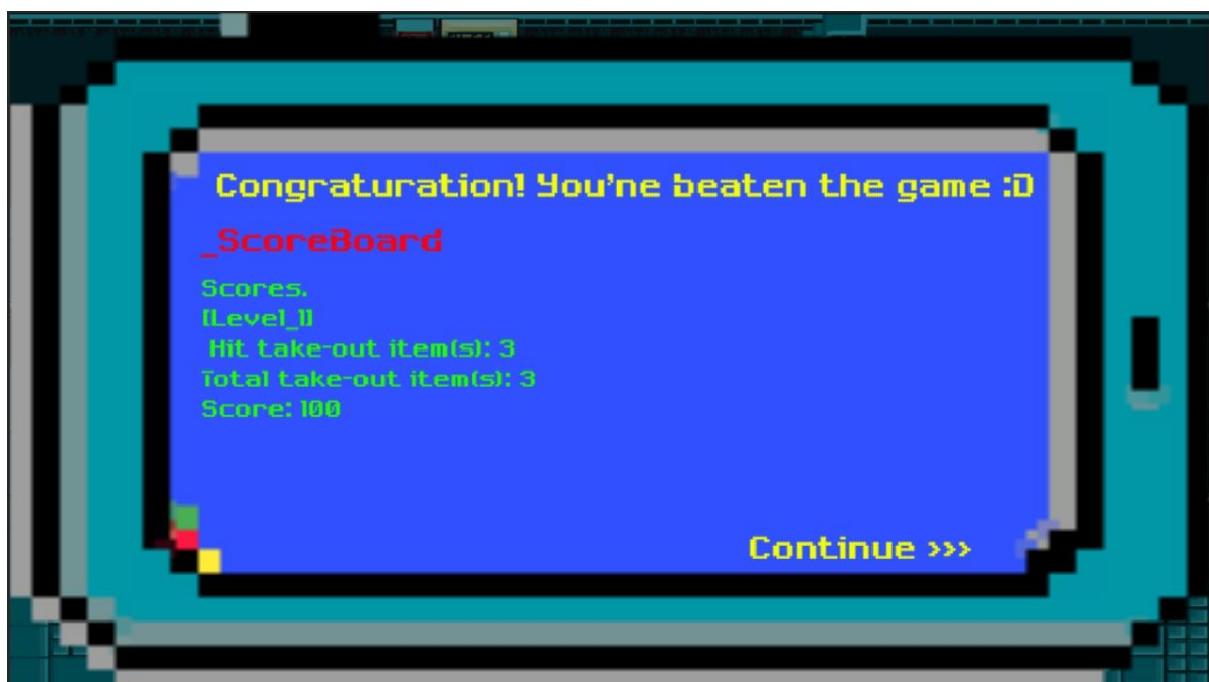
#Gameplay level 2



#Level win menu



*#Level lose menu*



*#Victory menu and scoreboard*





*#Credit*

## References

### Sprites

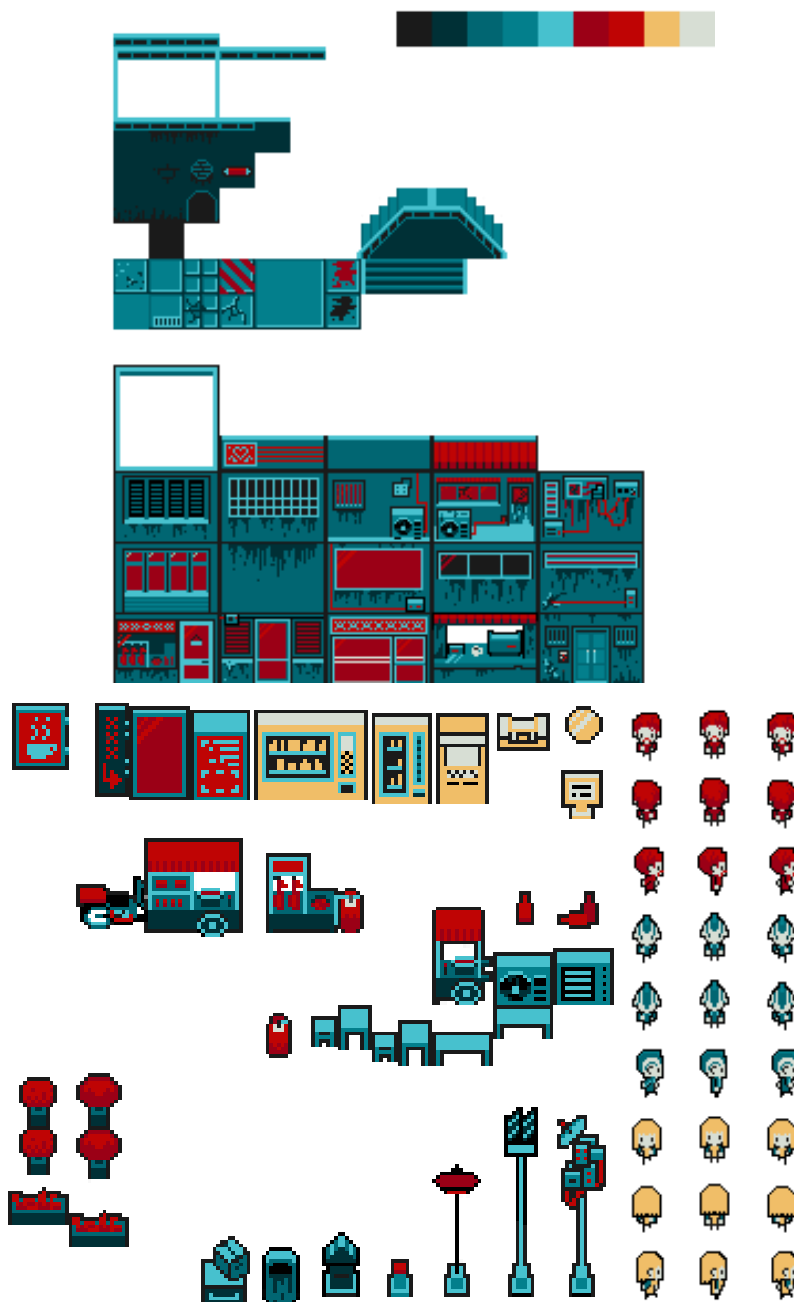
Neo Zero Cyberpunk City Tile-set

Author/Contributor: Yanin

Type: Open resources

Link: <https://yunusyanin.itch.io/neo-zero-cyberpunk-city-tileset>

Use: Map and NPC(s)



*#Game asset make use of- Sprite: Neo Zero Cyberpunk City Tile-set*

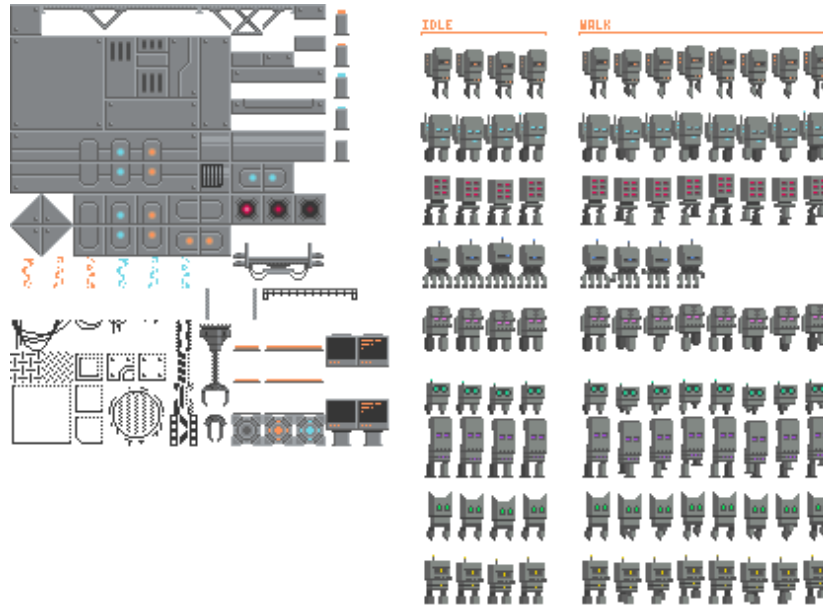
16x16 Tile-set + Robot

Author/Contributor: 0x72- Robert-

Type: Open resources

Link: <https://0x72.itch.io/16x16-robot-tileset?download>

Use: Playable character



***#Game asset make use of- Sprite: 16x16 Tile-set + Robot***

## Music


### Animal Crossing: City Folk- 2 AM


Source: Animal Crossing: New Horizons

Publisher: Nintendo

Link: <https://www.youtube.com/watch?v=YsDM3UvXWOc&t=22s>

Use: Background music (BGM) for the main menu, scoreboard, and credit menu

 Animal Crossing - City Folk...

 Animal Crossing - City Folk...

***#Game asset make use of- Music: Animal Crossing: City Folk- 2 AM***

### Animal Crossing: Bubblegum K.K. (Aircheck)

Source: Animal Crossing: New Leaf

Publisher: Nintendo


Composer: Kazumi Totaka


Main Producer: Katsuya Eguchi



Link: <https://www.youtube.com/watch?v=yYDMxwoTx9k&t=7s>

Use: Background music (BGM) for the in-game gameplay

 Bubblegum K.K. (Aircheck) ...

 Bubblegum K.K. (Aircheck) ...


***#Game asset make use of- Music: Animal Crossing: Bubblegum K.K. (Aircheck)***


#### Unlock new item game notification

Type: Loyalty free assets

Link: <https://mixkit.co/free-sound-effects/game/>

Use: Sound effect (SFX) for the order request notification

 Order Notification.wav

 Order Notification.wav.meta


***#Game asset make use of- Music: Unlock new item game notification***


#### Arcade retro changing tab

Type: Loyalty free assets

Link: <https://mixkit.co/free-sound-effects/game/>

Use: Sound effect (SFX) for the button clicks

 Click Sound.wav

 Click Sound.wav.meta


***#Game asset make use of- Music: Arcade retro changing tab***


#### Sci Fi positive notification

Type: Loyalty free assets

Link: <https://mixkit.co/free-sound-effects/game/>

Use: Sound effect (SFX) for the dummy robot system- *bag log*- sound

 System Sound.wav

 System Sound.wav.meta

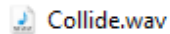
***#Game asset make use of- Music: Sci Fi positive notification***

#### Arcade collision

Type: Loyalty free assets

Link: <https://mixkit.co/free-sound-effects/game/>

Use: Sound effect (SFX) for the dummy robot collision with borders sound



Collide.wav



Collide.wav.meta

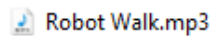
*#Game asset make use of- Music: Arcade collision*

### Toy robot walking movement

Type: Loyalty free assets

Link: <https://www.storyblocks.com/audio/stock/toy-robot-walking-movement-bxr-kzt2lwhk0wxwhqz.html>

Use: Sound effect (SFX) for the dummy robot movement sound



Robot Walk.mp3



Robot Walk.mp3.meta

*#Game asset make use of- Music: Toy robot walking movement*

## Scripts

### Shadow Caster 2D on Tile-map

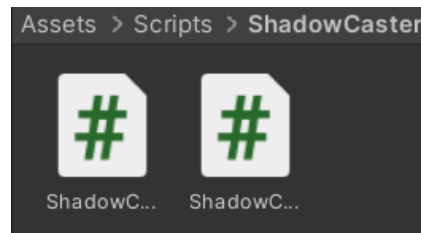
Author/Contributor: SeksitSeeton

Type: Open source

Link: <https://github.com/SeksitSeeton/Auto-add-Shadow-Caster-2D-on-TileMap>

Use: Implement shadow casting on the tile-map





### *#Game asset make use of- Scripts: Shadow Caster 2D on Tile-map*

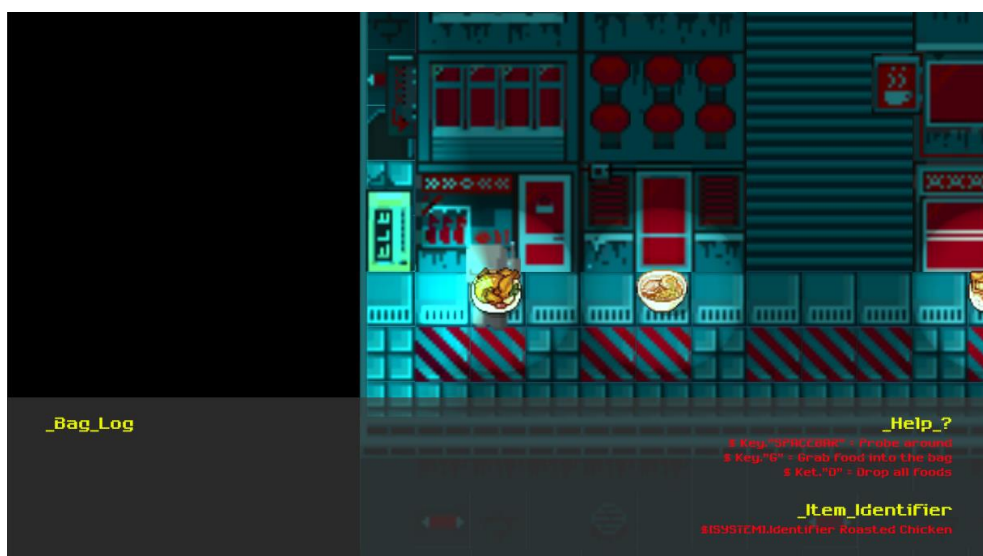
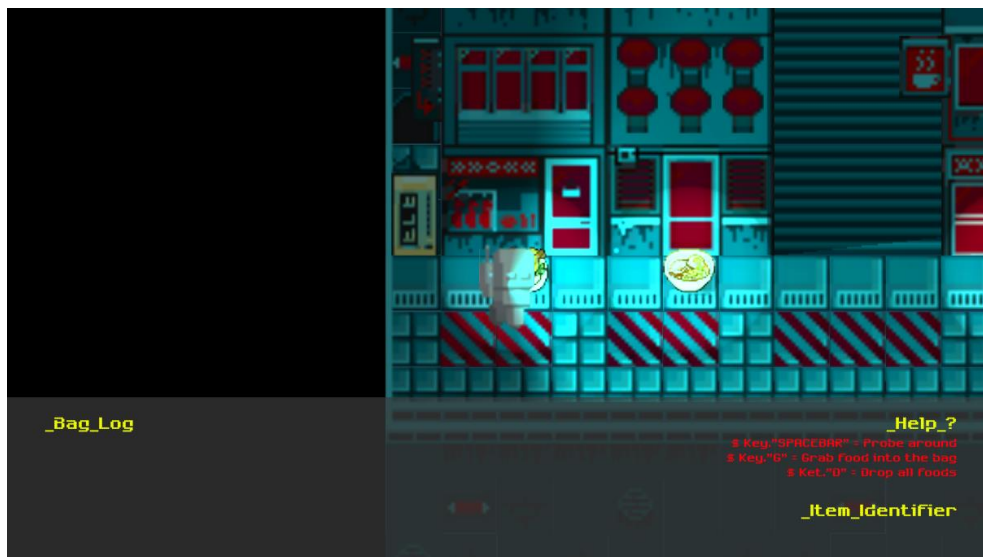
#### Real-time sprite sorting

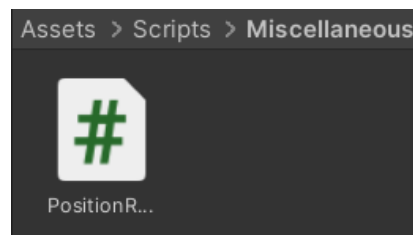
Author/Contributor: Code Monkey

Type: Open source

Link: <https://www.youtube.com/watch?v=CTf0WjhFBx8&t=249s>

Use: Implement the real-time sorting of the sorting layer of sprite





*#Game asset make use of- Scripts: Real-time sprite sorting*

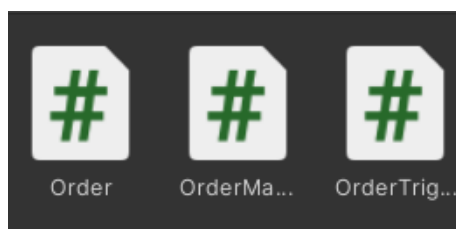
### Dialogue System

Author/Contributor: Brackeys

Type: Open resource

Link: [https://www.youtube.com/watch?v=\\_nRzoTzeyxU&t=803s](https://www.youtube.com/watch?v=_nRzoTzeyxU&t=803s)

Use: Referenced and modified to adapt into implementation for order request system



*#Game asset referenced- Scripts: Dialogue System*

**Project Product Demonstration- YouTube video presentation  
link-**

<https://youtu.be/bKIqyVojC7I>

## APPENDIX 1

## MARKING RUBRICS

Component Title	Final Project					Percentage (%)	40
Criteria	Score and Descriptors					Weight (%)	
	Excellent (5)	Good (4)	Average (3)	Need Improvement (2)	Poor (1)		
<b>Creativity of Game Idea</b>	Game idea is original, creative, and interesting. Students work to create something unique. Exceptional effort was invested by the student to generate concepts or potential solutions.	Game idea is original, creative, and interesting. Students work to create something good. Good amount of effort was invested by the student to generate concepts or potential solutions.	Game idea is acceptable, and interesting. An average effort was invested by the student to generate concepts or potential solutions.	Game idea needs improvements to make it more interesting. The invested effort needs more improvement by the student to generate a good concepts or potential solutions.	Game idea is poor. The invested effort is not sufficient by the student to generate a good concepts or potential solutions.	10	
<b>Game Characters (Playable and Non-Playable)</b>	Demonstrate excellent characters in the game that give a concrete form and personality to the ideas presented by game developers.	Demonstrate good characters in the game that give a concrete form and personality to the ideas presented by game developers.	Demonstrate average characters in the game that give an insufficient tangible form and personality to the ideas presented by game developers.	Demonstrated characters needs improvement in the game that give an insubstantial form and personality to the ideas presented by game developers.	Demonstrated characters are poor in the game that give an insubstantial form and personality to the ideas presented by game developers.	5	
<b>Music</b>	Demonstrate an excellent and appropriate music background and effects.	Demonstrate a good and apt music background and effects.	Demonstrate an average and partially apt music background and effects.	Demonstrated music background and effects needs more improvement.	Demonstrated music background and effects are not suitable.	5	
<b>Complexity</b>	Demonstrate appropriate, excellent game techniques in programming codes.	Demonstrate a good Game techniques in programming codes.	Demonstrate an average Game techniques in programming codes.	Demonstrated game techniques in programming codes needs more improvement.	Demonstrated game techniques in programming codes are poorly executed.	5	
<b>Game Fun, Enjoyment and Engagement</b>	Demonstrated an excellent good game to exhibit fun, enjoyment and engagement.	Demonstrated a good game to exhibit fun, enjoyment and engagement.	Demonstrated an average game to exhibit fun, enjoyment and engagement.	Demonstrated game needs improvement to exhibit fun, enjoyment and engagement.	Demonstrated game poorly executed fun, enjoyment and engagement.	10	
<b>Game Production Quality and Written document</b>	<b>Game Production Quality:</b> Product/Game prototype demonstrates considerable flare, works well and was demonstrated successfully. <b>Written document:</b> The sentences are clear and excellent grammar, spelling, syntax and punctuation.	<b>Game Production Quality:</b> Demonstrate sound good ability in creating the final product/game prototypes. <b>Written document:</b> The sentences are clear and excellent grammar, spelling, syntax and punctuation.	<b>Game Production Quality:</b> Demonstrate poor workable final product. <b>Written document:</b> The sentences and grammar, spelling, syntax and punctuation are unclear.	<b>Game Production Quality:</b> Product/Game prototype not fully working and did not contribute much to the demonstration. <b>Written document:</b> The sentences and grammar, spelling, syntax and punctuation needs improvement.	<b>Game Production Quality:</b> Product/Game prototype totally not working and did not contribute to the demonstration. <b>Written document:</b> The sentences and grammar, spelling, syntax and punctuation are incorrect.	5	
<b>TOTAL</b>							<b>40</b>

**Note to students: Please attach this appendix together with the submission of coursework**