**Medusa Hill Feast**

**– A Learning of Unity’s Fundamental Elements**

**Project Report**

**By Lin Xu**

**Instructor: Professor Michael Lajoie**

I. INTRODUCTION

*Medusa Hill Feast* is inspired by game *Stardew Valley* or *Harvest Moon*, it is a simulation game about running a restaurant in a fantastic world. Though player will only experience the life in Medusa Hill, player still can see what is going on in the world, also can be affected by it.

The full product will have full story line as well as the other plots about develop relationship with other folks in town or experience festivals, though in this project, only a demo, or tutorial will be done, which might need a lot of improvement to be a ready-to-ship product. I will try my best to catch some progress.

In *Medusa Hill Feast*’s demo, or my full project, those following are what I expect.

There will be a stable camera setting, which will follow the player to move around, and limited by player’s perspective.

Player control will be needed as will, which is allow player to let their avatar to move around the map and interact with other objects. While moving the avatar, the animation on avatar will be implemented.

Player also need to be able to talk with villagers in town, and there should be a proper panel to display it.

For player to walk on the map, a proper map is needed as well, with object collision and proper trigger for the event related with the object.

Plot will be needed as a part of a RPG game. In the demo, there will be some simple plots to explain the background story for the main character and the basic of the world setting, also to build the theme. The plot will be used to show some characteristic for the villagers or the main character as well.

A system to justify the relationship for the main character and the villagers will be needed, and to improve the relationship with villagers, the player will need a gift system, so they can send gift to the villagers. The relationship with villagers will give player extra benefit. In the demo, there should be a simple tutorial on how to send gift to villagers.

An interesting cooking system will be an important part of the game, still need to figure out how should player do the cook, so it could be more interesting without being too complicated.

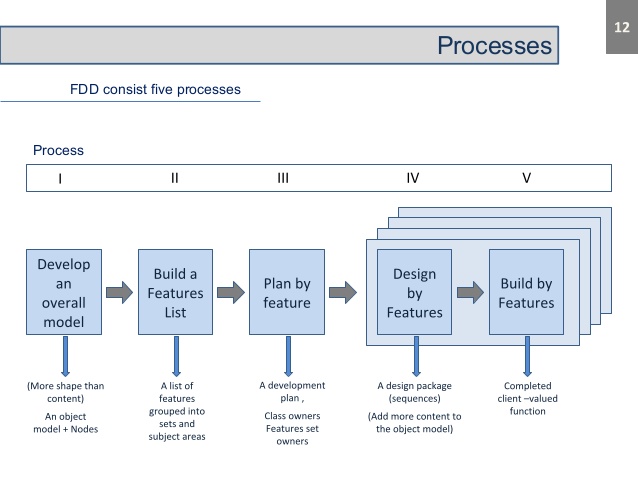
Besides the cooking system, a quest system so player can have goal in the game is needed. Player need to be able to receive quest, and check the progress, and finish the quest to get reward or reach the next part of the plot. The quest system must be tied up with the cooking system, so the cooking system is not too isolated from the other part of the game.

There should be different ways to gather the ingredients for cooking in the game, however, this project will not present them. If time allow, there will be a tutorial on how to trade with the store, use currency to get item.

The game design document has the story line for the demo project, and more introduction for each character as well as the background’s basic setting. The story will be very simple in the demo, just to show the understand of creating dialog for Unity.

In the project, what I did is using feature-driven development methodology. Before starting the project, I listed bunch of features on what I was planning to do, and plan on what does each feature needs when the schedule reaches that feature.

In the project, I will use Unity to produce a windows platform game, as my first time to use Unity, there will be lots of technique for me to learn, as well as the project building methodology. I am using Spiral methodology since I will be the only team member in the ‘team’. The worst cost through the whole project would be failed to complete the goal in my demo project before the deadline.



Before really starting the project in Unity, somehow, I decided to finish the drawing of all assets before any coding, and than can just apply all the assets when I start, instead of using any templet assets. I set a time line in Gantt chart to keep track on what I am going to do every week, and drawing assets was the second week’s schedule. After first week’s job done, which is making the entry screen. Though start a project with making UI first is not a good option, but in the first week I start to have some idea of what is using Unity to make a game project feels like.

Then I changes the timeline, move the drawing part until later in the project, and if I cannot finish the basic, I will not use own asset in the project, instead, only template assets from online free resource will be used.

To learn to make Unity project, there are lots of useful free resource that I can find, two major learning resources that I was using are book and YouTube video, all the resource I use will be append in the end of the report. There are also useful open source tools to supply Unity project’s producing. I was using Tiled to make tile map for my project, it is a vary handy tool that allow user to ‘draw’ the tile map using every sprite, and Tiled itself also has the function to add object collision onto the tile map. However, unity does not support Tiled file. There is an open resource program Tiled2Unity that can transform the Tiled file into Unity assets.

There are lots of things to learn when starting at Unity, and my main goal in my project would be learning Unity and if I could, also want to improve my drawing skill.

II. RESEARCH

In the project, most research I did is about using Unity with its build in functions. There are good tutorial books, but some of the code or functions from the book only suit for the older vision of Unity, which are not adapt with the new version. There are also some out-date functions online.

Unity is a popular game producing tool, and it is also easy to use, so there are lots of open resource tools or tutorials online that can be find easily, also, the game genre that I choose is popular as well, most functions about how to develop a RPG with fully explanation can be find online.

I had used RPG Maker before as well, and those functions that RPG Maker provided gave me lots of hits on what should my project contain.

In terms of researching, I did most of my study for this project from YouTube videos, and when I want to solve problems, Unity Documentation and Stack Overflow are helpful too. I read book Unity Game Development Blueprints and Mastering Unity 2D Game Development. Mastering Unity 2D Game Development has gave me a good starting point, however, books’ information is not as straightforward to learn compared with video tutorials for me.

Videos are helpful, but some videos providers have bad programming habit, which leads to having lots of inefficient code and weird code structure, it caused me some confusion when I was studying earlier.

III. PROBLEMS AND ISSUES

There were not lots of problems that I really cannot solve in this project, since all those elements I mentioned are just to learn the basic functions in Unity, learning the property of the needed function would let me manage through. Most of the problems that I faced are in some way related with the reference from the object to script.

There are some problems that I faced without related with refence, and here are three of them which I kept note on what was going on.

3.1 Problem with Menu Open and Close

When I was coding for the menu to be able to open and close using the same key using script MenuShowAtKeyPressed class. In this scrip, function Update() is called once per frame to check if the button Escape (which is called “Menu” key in the input system provided by Unity’s default input system setting) is pressed, if so, the menu will show up, and if the menu is already enabled, the menu will be set to disable to make the purpose of open or close the menu with a same button.

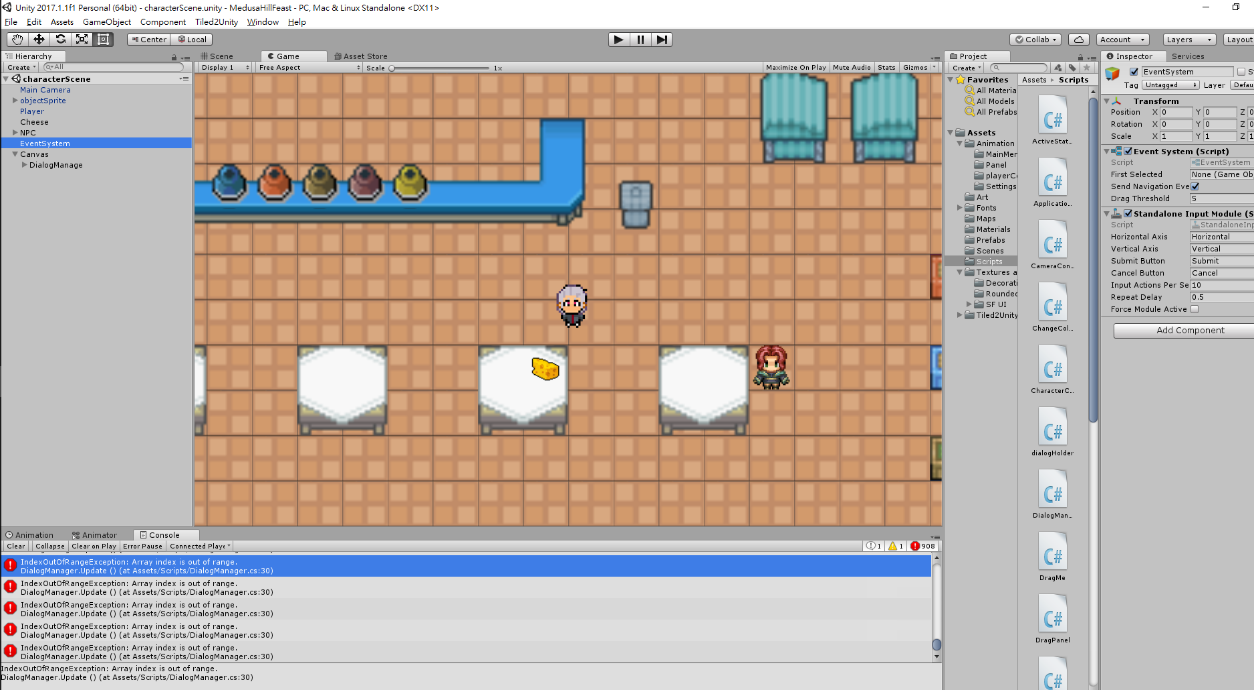
Update() function has if statements to manage the menu open setting, and there is a Boolean value openMenu to check if the menu is already opened.

I chose to use Input.getButton() function to trigger the if statement, and to manage the event. Somehow, in the test, the debugger told me that cannot find key “Menu”, since the input manager does not have it assigned, I checked the input manager, noticed since I changed the name of the key, it cannot use its default value anymore, so I changed the key name, and got it fixed.

After fixing that, there was no more alert, but the menu still would not show up, meanwhile in the log of the debugger, I can see that this event has been triggered for multiple times. This is caused by the trigger was set for multiple times so the openMenu value has been modified multiple times as well, and finally been set as the disable value. So, the code has been changed again, instead of using if-else statement, so this simple problem can be fixed.

This time I can see the menu by pressing the escape button finally, however, it flashed, because open or close are trying to active at the same time. The menu can be enable or disable, but it depends on how long to hold the escape button. Input.getButton() function caused the event to be triggered multiple times. Change Input.getButton() to Input.getKeyUp() fixed the problem, because unlike Input.getButton() to get the input every frame while playing, Input.getKeyUp() only be triggered after the key is pressed, when it pops up, during the process of press and release, it is only triggered once.

The problem got solved, but since Input.getKeyUp() required hard code for keycode instead of a key name as reference, player custom setting will not be able to apply for the menu control button.

 3.2 IndexOutOfRangeException for DialogManager

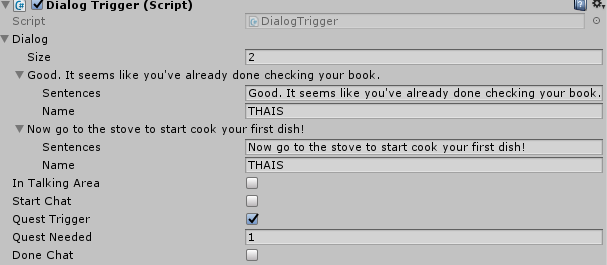
This problem is from after I set the conversation system, and trying to create new conversation with the same character. So, I duplicated the conversationTrigger, and rename them as conversationTrigger1 and conversationTrigger2. For each trigger, there are different dialogs for them to use implementing the serializable Dialog class.

When I test the new conversation trigger, and the exception happened, still the dialog from conversation trigger 2, but when reach the end of the dialog list, it still tried to go further and gave me exception.

Before this problem shows up, I was using the same structure with the tutorial video about building a proper dialog in Unit, it is using static value DialogManager to associate the DialogTigger with DialogManager. I realize that this link is probably not reliable, and so DialogTrigger and DialogManager are all messed up since they were all trying to link use the same static value while the static value also has to reference back.

So, I gave up on using static value for DialogTrigger to reference DialogManager, instead, there should be a public DialogManager named DM, and the developer will use this value to start a reference, not just relay on a static DialogManager value. Though set the DialogManager as public will cause more reference to developer to drag and drop in the interface, and increase the risk of wrong reference member in future, it will be easier to recognize, and debug compared with a private static value which can always out of control.

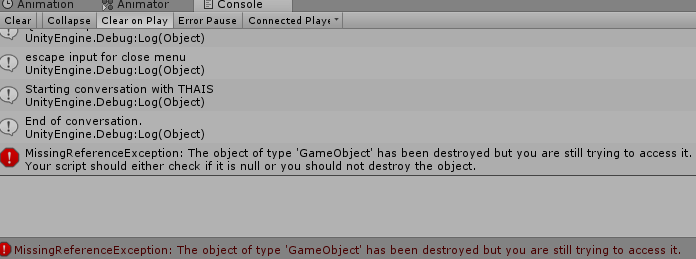
3.2 Problem with new quest accepting

 When I was trying to build the quest system, there are lots of problems to concern, and here is the one that has given me lots of troubles.

After the first quest is tested, and works well, I added another quest in the quest list using the same function, only changed the quest stage to NOT\_AVALIABLE since the second quest should not be triggered until the first quest is done, and need to talk with Thais to trigger it. So, there should be a trigger attached to the second piece of the conversation so like conversation 1 is able to trigger and accept quest 0, conversation 2 should be able to trigger and accept quest 1 as well.

After I set all the values properly and the references, quest 1 still cannot be triggered, instead, after the conversation, quest 0 is still being asked every time.

Therefore, I decided to destroy the old conversation trigger, so it would not mess up conversation trigger 2, it did not fix the problem, instead, it was giving an exception, tells me even I try to delete the reference, it will not try to reference something else.

So, I modified my code and re-code some of the code from DialogTrigger, and I believe that was the cause since it is the part which is directly linked with triggering the quest.

The problem was still not solved, DialogTrigger for conversation 2 will always trying to accept quest 0 even the triggered quest is already set as quest 1. ­­

The association between DialogTrigger and QuestManger was messed up. I find out the problem is because the link is tied together which cause DialogTigger always trying to access from the DialogManager, and DialogManager has a reference linked with conversationTrigger 1. After figuring out the problem, I set the link to associate between DialogTrigger and QuestManager for each pair, and set the public value in the interface to make sure there is no null reference. Therefore, the problem is solved.

Those are all the problems that I kept the record, besides those, there are still lots of minor problem about coding and interface, most of them are related with referencing in Unity, and some others are for Input Manager or sprite editor or Rigidbody2D or Scene Manager etc.

Those tiny problems I faced took me lots of times, and some of them even pushed the goal of the week onto the next week.

After I decide that I am going to finish modifying the project, and use this version as the final version, there is a new bug that I noticed, which is for the conversation, after it be triggered, the first dialog would show up twice. Since there is no animation between switching dialogs, this problem was never noticed before, and it can be fixed easily by modify the start count in DialogTrigger to 1 instead of 0. A problem for the dish button listener is the same, those problems were fixed.

There are some problems left in the project which I did not fix, and its about layout. The dialog in the dialog UI and buttons are squished to much and hide information when the application user set the application window too small, which should be improved in future.

So far, all the bugs I detected have been fixed, however, this project still need more features.

IV. ACCOMPLISHED

There are lots to be done in a simple little project like *Medusa Hill Feast*, I did not reach my original goal. The result looks so poor that I felt I could make this project in under an hour using RPG Make, however, I still learned a lot from this project.

In the first week, I learned about UI to start to learn Unity. This gave me some basic idea about using Unity and code with C#. Most of the coding are not quite different compared with Java, C# is quite easy to understand in the term of using it in Unity. The challenge is, associate the code with the program and the assets in Unity. I used template assets and was planning to change them into customized assets at the last week of working. However, there are too much of coding left and some bug to fix, there was not quite enough time for me to create them by myself. I will still replace some of them, but not all of them.

In my original plan, I was also going to create my own background music and implement it. The background music may still be added into the project from open source, so I can learn about how to implement music into the Unity project.

After learning about making basic UI and how to switch from scene to scene in Unity, the next goal for me is to implement the player figure, and let it has animation which match its movement. But before that, I noticed there are things I missed, which is the ground to let player walk on it. At this point, I started to learn about tiled map in Unity. After learning about tiled map, and a proper map is made. About how did those happen I will describe them in Part 3 and Part 4. In this section, I will focus more on what have I done in my project.

After a proper map is made, finally I can start adding figure movement into the project. Add a player avatar is quite simple, which is set a sprite into the project, and set it as a different layer compared with the ground and other objects on the map, while make it move involve with coding. Along with the coding, also need to set variable to ensure the facing direction of the player, so I can use it to create the animation using blend tree animation.

When the most basic is done, I can add more elements into the project now. Which is a NPC, and player should be able to interact with it. In the case of *Medusa Hill Feast* demo, Thais would be the character to have some conversation with player. This section would be the only part that I felt there is still a tinny bit of gameplay in this project, which is because I did spend some time to develop about the world background story, and so on, create the walking so player can get the most basic information of the world.

When talking is achieved, player now can accept quest from the NPC. In this project, for the simplistic, and in learning purpose, I did not set the quest as done when talking, instead, player has to open the menu, and open the quest panel, click the quest, and then click complete quest button to complete the quest. There are two small quest in the demo, which is open the menu and trying the cooking system.

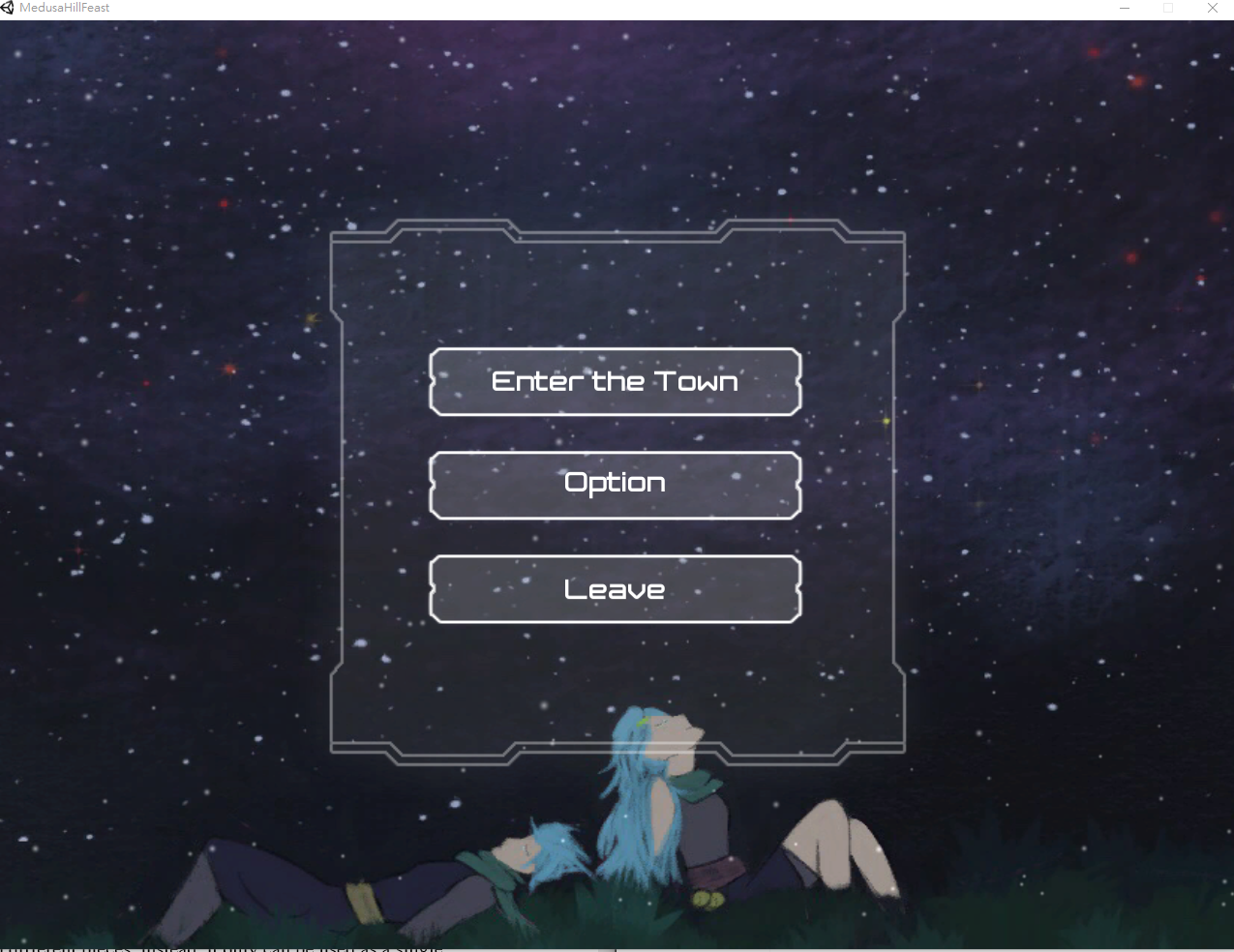
After finish quest system and debugging it, there was not much of time left for me to do the rest work, and originally, I thought this could be done in several pieces of code, which is only to trigger a tinny pop up window to show that the play did cook. My earliest plan, which is to make the cooking into a developed system, which is to open a new panel and add more gameplay and mouse onClick() event. I totally gave this one up since it would be too much for me to finish the work before the deadline. Though I have gave up on develop a complicated system which allow the player to do further interaction, cooking still cannot done by simply several codes. For user, this can be done in simply walk to the position and choose what to cook, to accomplish this, I will have to make a new panel and implement the buttons, so player can click, also, cooking requires ingredients, this should to be check as well. To see if player has it or not, player also need an inventory which can hold items. Therefore, the item system is created, and there is also a backpack in the project which is associated with players, so the whole project system, or in this case, just cooking system can know what items or ingredients that player has, and able to access it (remove or add). Player is not able to see what is in the backpack now, though there is a backpack window in the menu, it is not implemented to have any function to check what currently are in the backpack. Cooking system will check if player has enough ingredients, if enough, will remove the ingredients from backpack and add the dish item into the backpack, after finish cooking, where will be a popup window in the cooking panel to tell player that the cooking is done. If player does not have enough ingredients, the popup window will also notify that there are no enough ingredients. And that is the last quest, after player finish the quest, they can click “Leave the town” button in the main menu to leave the game.

Here are the details about what have I done, and learned.

4.1 UI Develop

To accomplish all the goals that I want in this project, there are lots of things for me to learn in Unity.

First, to create a usable user interface, and start using Unity, I learned about the canvas, the panel, and some other UI features. To set them proper, there are also canvas scaler and all kinds of layout or layout groups that I must know the basic to make the UI better looking.



There are still problems on customizing the camera for me, but there is no further need for me to learn more about camera setting in this project, that would be one of my future problems. To let the UI functional properly, coding is needed as well, so I learn about the basic of coding in C# with Unity, which include void Start() and void Update() functions, also how to use the buttons.

There are lots of useful pre-set of the button that developer can use to make the develop easier, such as can associate the button with a game object to set its activation by click the button. In the case of developing a user interface, I used the button to switch between the panels, when in the main menu, click option button to open the option panel, when in the option panel (set activation to false for the main menu panel, set activation to true for the option panel), click back button to go back to the main menu (set activation to true for the main menu panel, set activation to false for the option panel). In the user interface, there are still functions to switch between scenes, and onClick() function which is used for “Leave the town” button to terminate the application.

As I develop the user interface following the online tutorial, I also used the same assets as the tutorial does, which was download from the Unity assets store. This asset does not really fit the theme of the game, so I will change the background picture.

This user interface development gave me idea on how to develop a Unity project, and where should I go, and most of the basics. After user interface development, the next step I went is develop a nice looking tiled map.

4.2 Map Creating in Unity

Unity has the function to make a fully functional tiled map, but it is hard to modify, and it gives lots of problems, such as the tiles are not laid properly, and even it does lay on the layer properly, it still left some black lines which I cannot fix properly.

To make a nice-looking map, those procedures are required. First, user has to load the sprite into Unity. The sprite can be loaded in two ways, or two different sprite modes. Those sprites can be set to multiple or single. If it is multiple, it means this piece of sprite contains lots of sprites figures, which is the most common way of use. If the sprite mode is set to single, it means this sprite cannot be edit into different pieces, instead, it only can be used as a single sprite. After set the sprite mode into multiple, user has to use sprite editor to split the sprite into whatever pieces that sprite sheet has, and the cut can be set individually or divide the sprite sheet into pieces in rows and columns. Such as the spite sheet has total of 300 figures, 20 per row, 30 per column. After finish setting the sprites, user can drag the sprite into the scene editor, drag and drop to make a platform and so on. The old sprite on the scene editor can be covered by the sprite which is put into the scene editor later, and cause an in-layering order. However, this layering is not reliable, so for each sprite, user can also set the layer of the sprite, and the priority of the layer. Those sprites are mostly hold in a layer group, so user can find them easier. However, a map is normally made by over 100 pieces of sprites, with all those sprites loaded, the Unity will be taking too much memory while working, and it is not efficient at all. Those probably can be fixed by the Unity tile editor, but when I was struggling on it, did not find a way to solve this problem.

To make the map work properly, collider is also required. In Unity, what user need is to create collider object, and place it on the map, to represent that this piece of map cannot crossed by player. In Unity, the default tile setting is also complicated when trying to add the collision to the object, the polygon collider is good, but to set it properly, it is too complicated, and every collide object has to be set individually. I used the Tiled to generate the tiled map, sprites using are not quite different, but way easier to use and set. It has most of the functions that the Unity default tile map generator need, and even better. Tiled is really good and easy to use, however, the tile this program generates does not fully compile to Unity program. At this point, Tiled2Unity is needed.

Tiled2Unity is an open source program which is used to translate a map which is generated by Tiled into Unity. First, Tiled2Unity has to be loaded by Unity, and then Tiled2Unity can read the Tiled map, and place it onto the scene editor. All of the Tiled function can be adapted into Unity in this way, and all the collision set in Tiled is adapted as well.

Create a map is done after those procedures.

4.3 Player Figure

After creating a map so player can move, player figure will be the next topic.

First step to do is to find a proper avatar sprite, since animation is required. The sprite sheet which contains all the figures needed for the movement is needed.

The sheet in this project has total of 12 sprites to represent each movement the avatar has. Before applying those sprites onto the project, the script to move the sprite is needed. Since Rigidbody2D has been set as one of the components of the player spite, in character movement control script, I can just simply call GetComponent< Rigidbody2D >() function to associate the sprite with the code. Unity has built in functions to support the movement control, so in the code I just need to use Input class to detect the input direction. Unity also has built in library Vector2 to move object in 2D environment, combine those two classes, after set a value as avatar movement speed, finally the spite is able to move. When the script to move the spite is done, the next step is to make animation.

There are four different animation were made for player’s movement, which are walking toward four different directions. To make the movement animation, in Unity, I need to open the animation panel and drag the sprite on the animation timeline. Upon each animation were made, an extra piece of spite is still needed for each direction, which is for the facing when avatar stops. Add the stop facing for the animation was confusing me for a while, until I start to learn to use animator blend tree. The blend tree does not require extra code to function between moving animation and standing towards certain direction after the motion for blend tree is set properly. However, the old script for creating movement need to be enhances, which is add the value to verify if the spite is moving or not, and I also used two different value to verify which direction did player face when player stop input.

4.4 Dialog System

After spent tons of time studying and debugging, this dialog system could finally be applied freely on any object in the project. I prefer to call it dialog system instead of simple conversation panel since this is reusable and can trigger other events in the game.

The dialog system I have made was made by four components, Dialog Box, Dialog class, Dialog Trigger game object and Dialog Manager.

Dialog Box, or dialog panel was added as into a new canvas called Dialog Canvas, I had supported three text field into it, they are representing the name of the person who speaks, the dialog contains and an instruction information. Add the support to fit a picture is easy, but since I did not have enough nice-looking picture, I skipped this part. Those functions were learned when I started to learn UI, nothing new really. However, later from that point I had to implement code to update its text towards to other parts of the dialog system.

The Dialog class is the basic element of the conversation system, this serializable class I made contains two strings, they are the speaker’s name and the sentence speaker say, which matches with the elements in Dialog Box (the extra one is just an instruction tag, its contain will not change), for me to add pictures of the speakers later, there will be a GameObject reference to reference to the spite that is going to apply onto the conversation. This class is like a unit, and use Dialog Manager, it will become the fundamental part of this reliable dialog system. It is serializable, so in Unity inspector, producer will be able to set its value. In this case, if the game object implements Dialog, it will have public string value speakerName and sentence for me to set, I can say that the speakerName is “Rudolf”, and sentence is “Merry Christmas!” and so on.

Dialog Manager implement Dialog class, and it is tied up with Dialog Trigger. In Dialog Manager, first, there are Text game object references to associate speaker’s name and dialog sentence from public value that player can set to this script, so producer can change the text field in the panel later. Dialog Manager has three functions, no Start() or Update() is needed. The three functions are StartDialog(), DisplayNext() and EndDialog().

In StartDialog(), the panel will be set as active, so player can see the panel and text it contains. This method will also disable avatar movement, or menu open. When this method is called, the dialog array list will start to display from its first element.

DisplayNext() method change the text to the next Dialog element contains, also check if it is the last array in the list, if it is, EndDialog() will be called.

EndDialog() method as the end of the conversation, will set active as false for the dialog panel, hide it from the player, enable the movement of avatar, and enable to open the menu. If this dialog is marked as a quest trigger, at this point, EndDialog() will also call the function from dialog trigger to and so on to accept a quest. In case of interact with the same collision area for multi-use, EndDialog() will deactivate one collision field, or in another word, dialog trigger(mostly current dialog trigger), and active another one. For the dialog trigger that has no concern for effect the others, those values will all be set to itself.

Dialog Manager cannot be called by itself, it all depends on the Dialog Trigger which will be talked about next.

Dialog Trigger is tied up with Dialog Manager, they all contain reference to each other. Dialog Trigger is set as a component on a collision area, when stay at this area, this trigger can be triggered any time under certain conditions, in the project, the trigger is triggered by press button E (which is set as the action button in Unity default input setting). This Dialog Trigger is mainly used to associate the Dialog Manager with player control.

4.5 Menu

Compared with the main title menu, this in game menu has more functionalities.

This in-game function is made by three separate panels, player state panel, cooking book, and quest panel. Player state panel only gives the model looking and basic information, it has the field for backpack, but it contains no further information so far, and other parts are hard-coding as well, player state panel has not much functionality.

Cooking book are all hard-coding as well, only to give player the most basic information, or an idea of what one of the cooking book pages suppose to be.

Quest panel will be talked about along with the quest system.

The menu also has the button to close the application.

4.6 Quest System

Quest system allows the player to accept and finish it, not getting any reward in this porotype however. It is one of the most important element of the game procedure, and it is also the game lead.

Quest system was built by four parts. Quest Object, Quest Manager, Quest and Quest UI Manager.

Same as the Dialog for Dialog system, Quest as a serializable class, it is the fundamental element for the quest system, it contains quest ID, quest name, quest state, quest require, quest goal, quest description etc. Quest state as an important concept of the game, contains those states: not available, available, accepted, completed, done. Those states are set as enum in the script, so it can be called easier by other objects.

A quest is not available if the player does not meet the require for it, such as does not have enough level, or has not finished the pre-chain-quest. It will become available when player is verified to be able to accept the quest. If the quest is accepted, the state of quest in the all-quest list will become accepted, and it will also be added into current quest list hold by quest manager. When the quest requirement has been reached, the state of the quest will be set to completed, after completing the quest from the quest panel or NPC, the quest will be gone from current quest list, player will not longer be able to check it, its state will be marked as done.

Quest object contains an available quest list, which is only applied by the conversation trigger collision field, which is to verify what quests can be triggered by the conversation or other quest trigger. It is going to be set on NPC, object, or quest board etc.

Quest Manger is set as DontDestroyOnLoad game object, since it will be needed through the whole game. I used Awake() to initialize the Quest Manager as not breakable effect from when the application starts to run. Accept quest, give up quest, complete quest, add quest items etc. those import concept of the quest are all contained by this class.

To finish the quest, it always need to achieve certain goals, this prototype is the same, and the progress for the quest is represent as quest item. Such as if the quest need to defeat 10 slimes, whenever defeat one slime, there will be one quest item called “defeat slime” added into current quest’s quest item count, when the count reaches 10, the quest can be completed.

I had write lots of unnecessary functions for the prototype in this section, and those functions are not tested yet.

Quest UI Manager is used to associate the quest panel with all that quest information, so players can have a clear view on the quest panel for their quest information. On the quest panel, there is a list which read all the current quests player has got, and set them as a button. When player click the button, the selected quest information will be shown. And with a quest selected, the complete quest button can be clicked, which will complete the quest if there are enough quest item.

4.7 Inventory & Item

Though the item in backpack list is not implemented by the player state, but backpack is doing its work so far.

Serializable class is defined for item, which has the item name, ID, description, and image. Those items can be used for backpack or required item and other purposes.

The backpack holds those items, its information can be read by other classes so to check quest progress, UI display etc.

4.8 Cooking System

Cooking system was meant to be interesting, and more complicated, however due to the time limitation, there was not much I had done.

The cooking system can be access at certain places. In the prototype, the place to access the cooking system would be besides the stoves (or pots, or jars). When player stands besides it (in action trigger area) and press button ‘E’ to interact, the cooking menu will show up.

In the menu, player can select the dish, and click to check how many ingredients are required for the dish. When the detailed information shows up, player can click “start cook” button to start cook.

In cooking system, only the most basic requirements have been checked before player get the dish, since cooking level is labeled, but the leveling system is not developed yet, the only limitation for player to cook would be ingredients. If player has all the required ingredients in player’s backpack. If so, those ingredients will be removed from player’s backpack, and add a dish back into the bag, and a popup panel will be shown to tell player that something did get cooked; if not, the popup window will simply say not enough ingredients.

V. FUTURE WORK

There are lots of future work that should be done if to build this prototype into a ready-to-shipped game, and as a demo of the game, the tutorial should be able to present most functions in the game, so player be able to know the basic gameplay.

To finish this tutorial, those should be done.

5.1 Player Customizing

In the tutorial, player should already be able to pick the character that they want to use, Jacco or Jacca, and player should be able to name their character with a customized name.

This name will be save as a part of a new prefab named “playerName”, and therefore in the plot, player’s name will be called instead of the default name.

Player should also be able to customize the keyboard functions in the game, and maybe add controller’s control as well.

5.2 More Functions about Dialog

This game will not give player too much instruction, so to create some immersive feeling to player, and also because the control for this game is quite simple, player do not need too much instruction to enjoy it.

However, there will be some instruction for the cooking system, since I plan to make the cooking system more complicated, different dish will require different cooking technique, which will need different tools or ingredients and so on, so instruction for this part is needed. Those instructions will be added into the cooking book.

In the dialog, there are still things to improve, the UI should have better looking with some new sprites, and a bit of the animation should be also added to the dialog system, so switching between the sentence does not seems too different.

There should be figures on the NPC or main character in during the conversation as well, to represent their appearance and expression. The simple animation about player jumping or moving should be implemented too, to match with the dialog.

There is also another function needed which can be triggered by the conversation event that I was noticed at the presentation, which is player should be able to pick up the item to add it into the inventory, or have further interaction.

5.3 Art and Music and Sound Effects

All the sprites that I am using now need to be replaced, including the figure and the animation for NPC and player’s avatar, and tiled map since those are open source images I found online, and there was not enough time for me to add self-created sprites, those sprites are also not fit the theme that I want to create. The background image (as the first version and short in time, it does not have lighten and shade, need to add more layers).

The UI’s art need to be changed as well.

Talking about music and sound effects, for sound effects I will simply get some open source sound effects online, and will edit my own music to fit the theme that I want to create. This music should be relaxing, cheerful and lightweight.

5.4 Relationship System

This should be one of the most important systems. There are lots to concern in relationship system.

After finish two of the quest to know the basic, main character should be able to send the dish just cooked as a gift to Thais, and Thais will talk with main character about how to send gift to villagers to improve the relationship with them.

Every character has their favourite, liked, neutral, disliked, and hated gift, send the right gift will improve the relationship with the character, and disliked gift will make a character dislike main character as well.

The relationship will also be affected by the quest and plot. There will be different choices in the plot, and some choices player makes could cause a villager to hate you forever. This relationship system will affect some of the daily talking between player and villagers, and the quest received, also some plot. When the relationship reaches certain level, the character with high relationship with player will send player some gift to represent the friendship with player, at the birthday that player chose, the character will send gift as well, develop relationship can also can unlock achievements.

For special characters, such as the merchant, if develop relationship with her, player will be able to get discount for some of the merchandises.

Marriage will be applied on certain characters, based on the gender of the avatar player chose, the choice of marriable character will be different. The gender of player’s will also have a bit effect on the story as well.

5.5 Inventory System and Currency

Inventory system include the portable storage for player and the importable storage. This project already given the basic structure for portable storage which is the backpack, and the importable storage, chest, can be just a simple limited as certain objects’ event.

To improve inventory system, the item list should be modified to handle a list of class, instead of public string values. A proper list of items and their sprites should be set as well.

Currency should be set as a part of the inventory system too, and there will be three types of currency, they are Sungum, Moongum, and Stargum. Those currency’s amount and exchange will be balanced and tested later in the medium fidelity prototype of the full game.

5.6 Trade

Player will follow Thais to the town, and met the merchant, the merchant will sell player some ingredients, this trade system and adapted UI will be needed.

5.7 Fishing, Fighting and Gather

In the tutorial, player will be able to access the Mushroom Forest, and in the forest, there will be some seasoning fruit that can be gathered by player. Those items will be able to use as ingredients.

The forest also has a river. Player can do fishing besides the river. In the fishing system, player will need a fishing pole object to start the event, and must besides the river. There should be animation for player to throw the bait into the river. When there is fish gets the bait, player will hear different sound effect and a label will show up for one second and then disappear, if player is fast enough to get through this step, a new UI would show up. In the new UI, player has to move the mark, so the mark can be as close as possible towards the target’s center, when the mark is closer to the target’s center, the energy bar goes up and if far from the target, energy bar goes opposite, as the energy bar fills, the fish will be caught, if times up, or the energy bar drop under 0, the fish will escape.

Fighting system is another way for player to gathering ingredients. When player is not in the safe zone which has no monsters, player will be able to ambush by monsters hide in the bush or see monsters and diced to touch the monster, those are two ways to start a fight.

After entering the fight, new scene shows up, which let player and several monsters to fight in a belt-scrolling map as the action part of the game.

5.8 Procedure between Get an Order and Deliver

Player should know the basic procedure between accept a quest from the quest board until submit the quest.

First, player will get the order from Thais or whoever, and collect ingredients. Based on the complete condition, player will have to deliver the dish to somewhere in town or to Thais.

5.9 Ending Condition

In the tutorial, ending condition of the game should also be provided.

As an open-ending game, there is no actual win condition and a happy ending for it, however, there is loose condition and a bad ending for it.

At the end of the tutorial, Thais will let player to rent an empty property of her as the kitchen and restaurant, and tell player the way to pay the rent. Therefore, in the game player will be charged every month until player buy the property from Thais.

If player is too poor to pay the rent, or let the balance down to 0, player will enter the bad ending, which is the main character leaves the town and goes back to the battlefield, after that, no one ever heard about him/her.

The tutorial should introduce all the features for the players to grab players’ attention, and those systems which support all the features should all be developed. Those systems may be improved later, but all the programing functions should be prepared already.

There are way more future functions compared with which this project already has, but I believe develop those functions will use about the same amount time compares with the time that I have already used for this project, since I am already getting familiar with the C# programing towards Unity for 2D RPG project, and most of the Unity functions.

VI. APPENDICES

There are other documents about this project:

dialog.odt – This document was used to keep track on the conversation between the characters, and at first, I was planning on using a method to read file and implement the string read from the file to the dialog system. However, the tutorial I checked allows me to modify all the conversations in the Unity’s user interface, and if I want to read file from the system, it means those conversation might be exposed to the players out of the game, and there would be difficult on finding the right piece of the dialog from the file. Using dialog class and edit the dialogs inside Unity’s UI would be a better choice.

Medusa Hill Feast gdd.docx – This is the game design document for the project. This game design document only has the earlier plan I had for this project, there are lots of things unfinished or changed.

Medusa Hill Feast Time Line.xlsx – The timeline’s Gantt chart for project designing. This timeline has been changed once since I decided to move the drawing part to later phase, which is different from the earlier plan.

presentation.docx – This is the presentation power point.

ProblemSolver.docx - This is the document for me to keep track on the problems I met and their solvation. For lots of problems I met did not add them into this note since if keep all in record, it would be too much for writing.

There are four pictures I draw for this project. All those pictures have no shade and lightened, it is for time saving, and I was trying to create a light feeling theme added by the art too, I am not sure if this choice is good for the game, but those pictures does not suit for the map, and I did not have enough time to draw the map sprite by myself, so I did not add those pictures into the project.

Jacco.jpg – The original design for the male main character.

Jacca.jpg – The original design for the female main character.

JaccaAni.jpg – The original design for the walking figure for the female main character.

All the rest sprite assets are in the MedusaHillFeast\Assets\Art folder.

Here are the tools I used:

<http://www.seanba.com/tiled2unity>

<http://www.mapeditor.org/>

Here are the tutorials I used:

<https://unity3d.com/learn/tutorials/topics/user-interface-ui/creating-main-menu>

<https://www.youtube.com/watch?v=Pk3GCgaNVTY&list=PLiyfvmtjWC_X6e0EYLPczO9tNCkm2dzkm>

<https://www.youtube.com/watch?v=Tm2L-_0eIeY&list=PLiyfvmtjWC_X6e0EYLPczO9tNCkm2dzkm&index=2>

<https://www.youtube.com/watch?v=Oao-A7bkve0&index=3&list=PLiyfvmtjWC_X6e0EYLPczO9tNCkm2dzkm>

<https://www.youtube.com/watch?v=M1bEtHOE8z8&index=4&list=PLiyfvmtjWC_X6e0EYLPczO9tNCkm2dzkm>

<https://www.youtube.com/watch?v=J6BQ4Fcy4cc&index=5&list=PLiyfvmtjWC_X6e0EYLPczO9tNCkm2dzkm>

<https://www.youtube.com/watch?v=sVCXgN5_XmY&list=PLiyfvmtjWC_X6e0EYLPczO9tNCkm2dzkm&index=6>

<https://www.youtube.com/watch?v=03-zRkJsoUw&list=PLiyfvmtjWC_X6e0EYLPczO9tNCkm2dzkm&index=7>

<https://www.youtube.com/watch?v=p1rONoZ6u8o&index=8&list=PLiyfvmtjWC_X6e0EYLPczO9tNCkm2dzkm>

<https://www.youtube.com/watch?v=x9lguwc0Pyk&index=9&list=PLiyfvmtjWC_X6e0EYLPczO9tNCkm2dzkm>

<https://www.youtube.com/watch?v=xFc6AsHMkK8&list=PLiyfvmtjWC_X6e0EYLPczO9tNCkm2dzkm&index=19>

<https://www.youtube.com/watch?v=BrCxFpZFRS0&list=PLiyfvmtjWC_X6e0EYLPczO9tNCkm2dzkm&index=28>

<https://www.youtube.com/watch?v=cVS9-nPmzGg&list=PLiyfvmtjWC_X6e0EYLPczO9tNCkm2dzkm&index=29>

<https://www.youtube.com/watch?v=KWNzLT46w9Q&index=30&list=PLiyfvmtjWC_X6e0EYLPczO9tNCkm2dzkm>

Mastering Unity 2D Game Development Published by Packt Publishing Ltd. Projuction reference: 2211014

(Chapter 2 Character building Chapter 3 Getting Animated)