The goal of this document is to encapsulate my thought process while working on ABG RUSH. Of course, not everything will be here, but I will be copying and pasting anything that I write down on sticky notes into this journal. That way I have an idea of what was going through my mind while I was working on a specific aspect of the game. This will also allow me to write down ideas I am either not able to implement due to not being in the scope of the project, or because I am simply not able to focus on that part of the project at the time.

June 1st

Each game object could set its own z value. This would require another script on each and every object in the scene. This script would simply set the z value = y value. This way the objects that are higher(y value) in the scene will appear to be behind objects that are lower in the scene (y value).

Since each object would be doing this themselves, this could cause some trouble with processing, but it shouldn’t be much, even if it is being done each frame.

Another option would be to have the games manager handle this as well. Each time a new object is created (sprite wise) it informs the manager, the manager adds it to a list, and each update, or specific frame (1/30) the manager will set the z values of each object to its own y value. This would still require each and every object to inform the manager of its existence, but since each object will most likely be communicating with the manager, no extra step is required on startup. Objects that don't move should probably still inform the manager and be updated each frame, or at least on startup. This will prevent changes in the editor, such as moving objects around from ruining anything.

June 2nd

Each object that the player can interact with will have a location for patients, and a location for players. So if the player interacts with a chair, the player will stand in front of it. If a patient on the other hand interacts with a chair, the patient should stand on top of it, and change it's animation.

All objects that the player can interact with should have 3 states. Idle, Active, and Hovered (which is a pseudostate).

Idle will have a very simple/normal looking sprite and no animation.

Active will have a brighter looking/more eccentric sprite, and possibly an animation. Either that, or the patient within the object shall have some kind of animation/change.

Hovered will possibly change the cursor icon, and change the objects sprite or animation.

Hovered can only be accessed if the object is both active, and the mouse is over it.

Removing the mouse or clicking on the object will turn it back to it's active state.

For the z-axis, only objects that move (nurse and Patients) need to actively update their y/z position. All others only need to set it once.

All interactable objects will be based off of a single class InteractableObject

Keep track of state : idle, active, hovered

Nurse Position Location

Patient position Location

So for instance, if the player clicks on the reception desk:

Player.setposition.receptiondesk.NLocation and the nurse will move to the reception desks n location.

Since I plan on making the waiting room chairs and testing those first, I have made animations for them. These animations are: Idle, Active, and hover. They are for the states that the chair can be in. So when the chair has nothing going on, it will be in its idle state. When it has a patient, it will be active and it will slightly change color. When the chair is hovered over, the chair will slightly rotate.

Before making these animations and any others that will be in the game, I have decided that objects that will be interacted with will have their specific sprite as a child component of themselves. So for instance, I made the chairs first, set up all the properties and components. I made sure that the collider fit the chair, and then I removed the sprite. I then created a child gameobject of that object and placed the sprite there. Due to this, animations will only incorporate the gameobject’s child sprite, and should never effect any part of the actual object. This will allow me to create animations without fear of anything else possibly being messed up. So for instance, If I made the hover sprite interpolate between sizes of normal and large, and rotate, the actual object itself, and it's colliders would not be changed or affected by this. So the hitbox will remain exactly the same, and not cause any problems for the player.

Appear to have run into a small unforseen problem with layering. Although I am setting the z position each update, it's staying/resetting to 0 during movement, and remains that way until the nav agent has stopped moving. This leads me to believe that I may have to change something within the navagent scripts to convert them from vector2 to vector3. This would allow there to be a constant z value instead of it being reset to 0 each frame when the vector3 position I create is beind downcast to a vector2, which is what appears to be happening.

After taking a brief look through the polynav code... it looks like more time may need to be devoted to this in order to make objects be drawn on top of each other in the proper order. I will probably ask warren to give me some feedback or ideas on what to do. i think I am done for the night. I'm going to copy and paste this to the journal, upload the repository, and upload the current version of the game to google drive.

June 3rd

Today I would like to...

Have 3 hotspots working - Reception, Waiting Room, Patient Room

This would include

- Location for nurse and patient

- Send both the nurse and patient the correct location data

- Become Idle/Active Depending on status

- Know if a patient is using this hotspot, so if a patient is currently in this waiting chair, and return the patient

- Have an active function such as clicking on an active waiting room chair, will make the chair inactive, and the patient should move to an open room if one is available

Manager

- Keep track of the number of open rooms

- Keep track of the number of open waiting chairs

- Get/Send the above information if needed. So being able to verify the number of open rooms

Nurse

- Move to correct location for each hotspot

Patient

- 3-4 different states that have their own specific timer values

- Move to correct location for each hotspot

- Tick/Countdown the patience timer

Creating a person class to be the parent of both Nurse and Patient. The main reason is for movement. If movement is handled through the person class, then I won't have to differentiate between calls later on. I feel like there should be more that the person class can do, but I can get back to that later once I have movement working.

I would like a way to differentiate between nurses and Patients, and I think tags may be the best way. This will allow the Person class to differentiate what it does. So for instance, when a location/hotspot has been reached, the Person class will inform the object to update it's status. A patient that was moving to the reception desk would therefore update it's status to match that. And a nurse on the other hand would open up the dialogue box for the first available patient.

Patient's should not continue to tick down their own clocks if they are being interacted with. So I need to create a check somewhere that will prevent this from happening. It can be in the patient class itself. The problem is, do I have a function called inside the patient that flips the switch or should it be done from outside? For instance, if the player clicks on this patient while their in a waiting room chair. The patients UI will pop up. At this point, the patient needs to stop ticking down its timer. And when this UI is closed, either by performing an action or inaction, the timer needs to either continue clicking, or remain stopped.

I feel like outside would be better. For instance the UI will know what action was made by the player, and can inform the patient of this action. So in this instance. UI Opens. UI informs patient to halt timers. If the player clicks "Treat", the UI closes, the patient's timer remains halted, and the patient moves. If the Player clicks "A room will be ready soon", the UI should close, the patient should have the pacification delay added to it's current timer, and the current timer should become un-frozen/halted. In both of these situations, I also feel that the UI should inform the current hotspot of what happened. So if the patient is moving elsewhere, the hotspot should free itself up, and change it's status.

At the same time though, a Patient may also need to be the one informing it's current holder/hotspot that it's leaving. So it may be better if the UI doesn't handle this, and only the patient does. This is simply because the UI would have to have more added onto/into it to handle that, while the patient already has to inform the hotspot due to the fact it can leave/storm out.

Should the ever present patient patience timer be a total amount of time or segments of time? And if it's segmented, should leftover time be added back to make a new total or should it simply be gone? The former would mean that the player has at least 50 seconds to get the patient in and out. The latter would mean that the player has only 15 seconds to sit the patient down in a room or in the waiting room. If this time elapses, the patient leaves. And the addition to segmented would mean if the player sat the patient down with 5 seconds left, do those 5 seconds simply get lost? Or do they get added onto the next timer. Also, if the patient is briefly pacified (more time added), does this extra time get added as well or no?

Only Exam Rooms, Waiting Room/Chair, and Triage actually use a/the patients. So should patient interactions be done individually or within interactableObject or should another parent/child relationship be created? I think I'll make another parent/child relationship. This will trim down what's in the top parent, and make it so that other objects, such as reference desk, sink, and more, have much less involved with them.

Those 3 will require 2 states, idle/active and Hovered. They do not need a patient variable since they will not interact with them. They will also not need a vector2 for the patient's location. They will each perform their own functions. I think I may re-write the Interactable Object Class, and the other few really quick because they are currently very small, and I would feel better renaming and repurposing them now, instead of possibly later. New Names will be:

OfficeObject - Objects around the office. This will encapsulate everything.

- Access to the manager

- Location for the nurse to stand

- Access the object's animator

- State Controller/manager for animations and actions, Individual classes should be able to set their state. So the object's script should determine if the object is idle or ready.

\*Going to stop using Active, and start saying Ready. Also, I started saying Exam Room instead of Patient Room.

- OnMouseOver / Exit, since each of the objects will have these, it would be better if handled from inside.

Patient Objects

- Child of OfficeObject

- Location for patient to stand/sit

- Return location for patient to stand/sit

- Patient variable for current patient.

- Return the current patient

- Add/Remove the current patient

- Set the status of Idle or Ready based on current patient (OfficeObject handles hover)

WaitingChair

- Child of PatientObject

- Patient sit animation

- UI Information/Data

Triage

- Child of PatientObject

- Queue up patients

- Patient Story UI

ExamRoom

- Child of PatientObject

- Patient sit animation

- UI Information

- Update Patient State (multiple Times)

Sink

- Child of OfficeObject

- Set the Nurse's clean bool to true

Reference Desk

- Child of OfficeObject

- Turn on the Computer / Show Help UI

- Pause Game?

After writing these up, the differences between them seem very negligible, but I feel that I should separate the classes anyways since more may be added on down the line.

Still need to queue up the patients that appear at the triage desk. Other than that, The patients currently go to an open slot/position,and will wait there until there time runs out. Then they will leave. Of course, it doesnt look pretty at the moment due to a lack of assets, and I don't feel like making some animations right now. But they will look better later. I need to add some kind of animation or a small sprite change to the hotspots themselves so I can determine what's going on during testing. The triage only appears to work if I click it on the left side, which is strange, but since I don't have a visual effect when I mouse over the hotspot, it will be difficult to know exactly where it is. Which is why I need to add one!!

I would say the majority of the goals I wanted to get accomplished for tonight have been reached, but I am currently too tired to verify each and every single one of them. I will be meeting with Professor Warren tomorrow so that will be a good time to bounce some ideas off of him and see what he thinks so far. I plan on having the majority of this movement shenanigans finished by the end of this weekend. This is definitely taking a bit longer than I thought it would, but I guess that's because one does not simply program movement. In order for movement to work, states had to be implemented, states required hotspots, and hotspots required different classes containing all sorts of information. I honestly feel like The majority of the most trivial portion of the coding will be done once movement is. And thats simply because movement required so many of the other aspects of the game to be at least started.