

2021-10-17 Meeting notes

Date

17 Oct 2021 10pm - 12am

Participants

- Avocado (@ Gabriel Ting @ Andrew Xie @ Prayag Rawat @ William Feng @ amandaliu120)

Goals

- Discuss UML design ideas with team
- Formulate initial design of project

Discussion topics

Item	Notes
<p>Previous Actions</p> <ul style="list-style-type: none">• Create UML design• Set up Lucid collaborative workspace - https://l.facebook.com/l.php?u=https%3A%2F%2Fucid.app%2Fucidchart%2F8ecae5c6-9728-46d3-b1f2-a608ca77df8a%2Fedit%3FinvitationId%3Dinv_607df093-6ebb-4da5-9e90-6ce242d1aa25%26fbclid%3DlwAR0eEhndPLC1xFLhDg5cyIT07uGgZ5dNqyorb6o35coUuKlj99mB1KOck9A&h=AT0ch3hrqX-QjUTNaU3MMrLicg_Gcid6R_I3S6l7vLWnuA79FSMMSaRmOdQkoMHjWrdtA5YhZtTTrWT8py_S5LuxTpaQRn62d2L5TiZhEB38cns7u6YzapeARC1C9n93qAVMvLXVcbT1GSFWRZIJig• Plan our weekly timeline• Zoom Meeting Link: https://unsw.zoom.us/j/81763180845	<p>Emails for reference</p> <ul style="list-style-type: none">• x.and.i.rew.e@gmail.com• williamfeng2001@gmail.com• gabrielting1@gmail.com• prayag1601@gmail.com• amandaliu120@gmail.com
<p>Story Points</p> <ul style="list-style-type: none">• Should we introduce story points? Relative measure of effort needed to complete a task/ticket - gauge distribution and complexity of ticket• Fibonacci scale: 1, 2, 3, 5, 8, 13<ul style="list-style-type: none">• 1pt - 1-2hrs• 2pt - 2-5hrs• 3pt - 6-9hrs• 5pt - 10-14hrs• 8pt - 15-20hrs• 13pt - 21-27hrs	<ul style="list-style-type: none">• Good for distributing tasks and ensuring that no one is getting overwhelmed• Probs unnecessary at the moment since we'll all be helping, working and putting functions together• May introduce later or for further milestones


UML Design

- Discuss everyone's initial design and approach to the project
- Consider the pros and cons of each
- Having a table with checkboxes so that we know which entities are movable /static/collectable/buildable
- Goal strategy arraylist so that we need to accomplish all of those goals before we can win the game (but how to account for AND/OR goals?)
- Strategy, composite, state patterns are most useful? Design by contract
- State pattern for the game modes - have modifiers for all the classes
- State/observer patterns for the mercenary (allied/enemy), and character (invisible, invincible) etc.
- Abstract class for moving entities potentially - since all of them have the health and attack damage attributes + battle interface
- Goals: composite/strategy pattern (each node represents goal comprised of minigoals vs handles easy list of goals) maybe even explore observer pattern so that we know when the game has ended?
- Mercenary: observer pattern
- Rather than having an arraylist for goals, we could also have an attribute within the subclasses for the subgoals?

Action items

- ☐ Next meeting time: Tuesday 10pm-12am
- ☐ Avocado (@ Gabriel Ting @ Andrew Xie @ Prayag Rawat @ William Feng @ amandaliu120) Continue with design and making it more detailed

Decisions

-  Established a rough outline of the design