Meeting Agenda

Group: Super duper Omega gruppen
Date: 2020-09-14 10:00
Chair: Erik
Participants: Samuel, Oskar, Erik, Behroz, Sebastian, Alex (from 11:00-11:45)

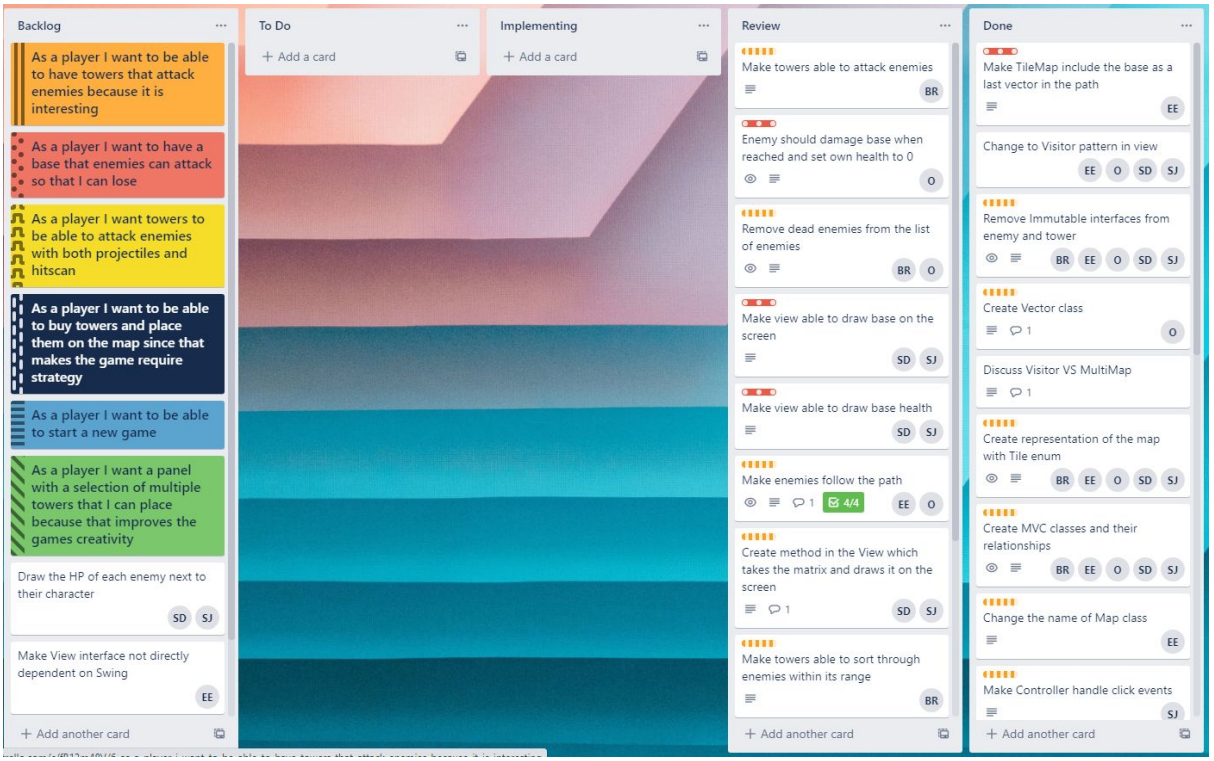
Objectives (5 min)

- Create RAD
- Add new tasks for projectile story and assign some of them

Reports (15 min) from previous meeting

Everyone removed the ImmutableTower and ImmutableEnemy interface.
Oskar, Erik, Samuel, Sebastian implemented visitor pattern.
Erik added a base tile in the map (and added it to the path enemies follow).
Sebastian and Samuel added drawing of the base and its HP.
Oskar implemented Health with a little help from Erik, as well as having enemies damage the base and get removed when they reach it. They also made the game stop when the baseHealth==0
Behroz made towers able to attack enemies within their range.
Oskar fixed a problem with Travis.

We finished all of our tickets in todo, currently our board looks like this:



Discussion items (135 min)

We decided that every enemy should have the hitbox of a circle, half tile in diameter. Also, every projectile should be a point (as for now at least).
Samuel, in charge of sprites, should make every tower face right to make the getAngle logical.

Alex Gerdes joins the meeting to discuss a few things.

He thinks that the UML class diagram has much duplication. The domain model is very important in this stage! Alex disagrees that the class UML diagram is more important than the domain model. The domain model should not have verbs. He thinks our domain model is good (lucidcharts bottom left).

Every functionality comes from user stories. Therefore there must be a user story that says they want both flying and walking enemies before implementing functionality that makes that easy in the future (changeable DefaultEnemy). Alex would take the easy road first, learn from what is good and bad about it, then make changes to it later.
Instead of having too many abstractions from the start (inside package e.g. towers), maybe create a facade [...]. Simplify. Figure out what information everything needs, *then* figure out how to do that. For example: the path the enemies take can be stored in the factory.
“Why does the TileMap know the path?” Think about different ways to implement things. The path that enemies need should probably be stored in the enemies themselves.
[Discussion about tile grid coordinates and enemies having floating point and tile grid coordinates]

How we show things and how we save data are two different things. **Watch out for having a too complex model just to make drawing easier.** Make it as simple as possible (think about removing services and handlers). Watch out for almost empty classes with no information. In this state the structure should be simpler, not having that many abstractions. Also, make the user stories drive the development.

Outcomes and assignments (5 min)

Make enemies own their own path instead of asking for it.
Make more user stories to decide **in detail** how the game should be in order to make deciding structural stuff easier.
No assignments because we need another meeting deciding more user stories first.

Wrap up

Next meeting is tomorrow 2020-09-14 at 10:00 or 15:00 depending on when the seminar is.
Behroz intellij is broken again, must be fixed before we code again.