**Facilitator:** Linda Evaldsson
**Participants:** Linda Evaldsson, Johan Swanberg, Hanna Römer, Simon Gislén
**Date:** 28/4/2015

# Meeting Agenda

Group meeting for group 11, Proximity

## §1 Objectives

- We need to add key bindnings
- We need to discuss the structure of the tower class
- We need to discuss how to implement user interfaces
- We need to discuss testing

## §2 Reports

We have started working on the report and finished the disposition and intent.

On the programming side the following things have been implemented:

Simon has:
- Built a framework of spells
- Added so creeps give XP on devolve that affect the player level
- Added so creeps give resources on devolve
- Started working on the user interface for level/xp

Johan has:
- Added art assets for bases, backgrounds and particle effects
- Remade  sound so it plays in left/right channel.
- Added four working spells based on the PersistentObject class
- Changed so an "add stack" is used instead of adding directly onto the global list
- Found several bugs that are added as tasks to Trello
- Bug fixes on creep, for example fix of the hit box
- Bug fix on path so creeps follow it correctly
- Tested intelliJ uml generator, it seems it only shows extends/implements associations.

Hanna has:
- Finished the pause game and speed control
- Started working on the main menu
- Added options to change volume

Linda has:
- Restructured the comments and added a few comments that were missing

## §3 Discussion Items

1. Keybindings

2. Should we restructure the logic of the tower class?

3. How can we better implement user interfaces?

4. Should we refactor the name of the class PersistentObject?

5. How should we refactor the folder structure of our classes?

6. How should the main menu look?

7. How should we divide the testing?

## §4 Outcomes and assignments

**Outcomes:**

1. We are going to use key binding for towers and spells. Spells will be bound to "Q", "W", "E", "R". Towers will be bound to the numbers 1-9.

2. We concluded that we need to restructure our Tower classes. We currently have an abstract class Tower with a lot of Shooting-logic. The other Towers extend this class. We need to instead remove some of the logic from Tower and add a ticking-function to this abstract class, so it can be used for other logic as well (not only shooting).

   We will add an abstract class "ShootingTower" that extends our new Tower class. Shooting towers will extend this class instead. ShootingTower will work approximately like the current Tower class does.
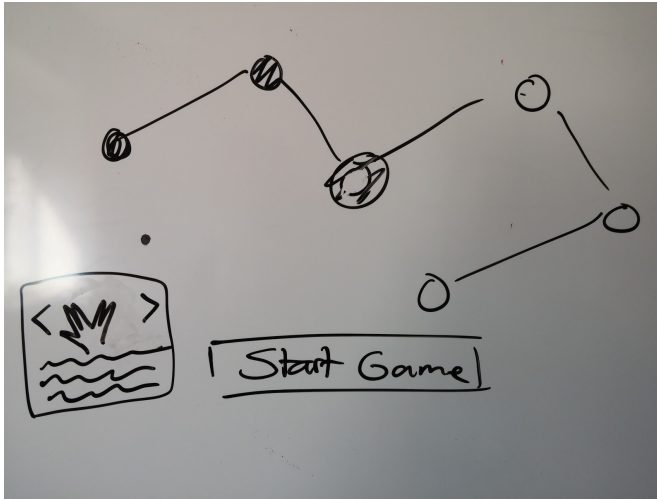
3. We will merge all the control panel controllers into one. The class that will be left is the ControlPanelController-class. The ClickHandler will be slightly modified to be able to handle multiple models per controller.

4. We will not refactor the name of the class PersistentObject. However it is currently only used for our spells and it is not logical to have the name "PersistentObject" for a list of Spells. Because if this we will add an abstract class "Spell" that extends PersistentObject. This will also make it easier for us to modify the "Spell" class without affecting all PersistentObjects, for example to add key bindings.
5. We need to have more packages and will create the following and possibly more packages under the current "Model" package: "Player", "Map" (for all objects on the board) and "ControlPanel".

6.  We used a whiteboard to draw up the main menu. See sketch below:



7.  We first need to set up JUnit in order to begin the testing.

**Assignments:**

1.  Linda will implement the key bindings.

2.  Simon will restructure the Tower classes

3.  Linda will merge the control panel controller

4.  Johan will add the abstract class Spell as discussed and also make this abstract class implement Holdable. He will also make some more concrete spells.

5.  Simon will add new packages as discussed

6.  Hanna will continue implementing the main menu

7.  Johan and Linda will set up JUnit together

# §5 End of Meeting

Next meeting is thursday 30/4 10:00 at group room 3217, EDIT, Chalmers