C++ Project Plan, Minebombers

Henri Nurmi (345545), Olli Rauramo (431433), Jere Vaara (294450), Max Huttunen (348283)

The Scope

We are planning to implement all the minimum requirements paying attention to quality. We are planning to add at least sounds and configurable keys as extra features, probably extending the project further with other extra features.

We would also be interested in implementing a multiplayer mode that works over network, but we are concerned about the time constraints. We will take a better look at this at the end of the project.

Selected extra features

- Audio
- Configurable keys

Potential extra features

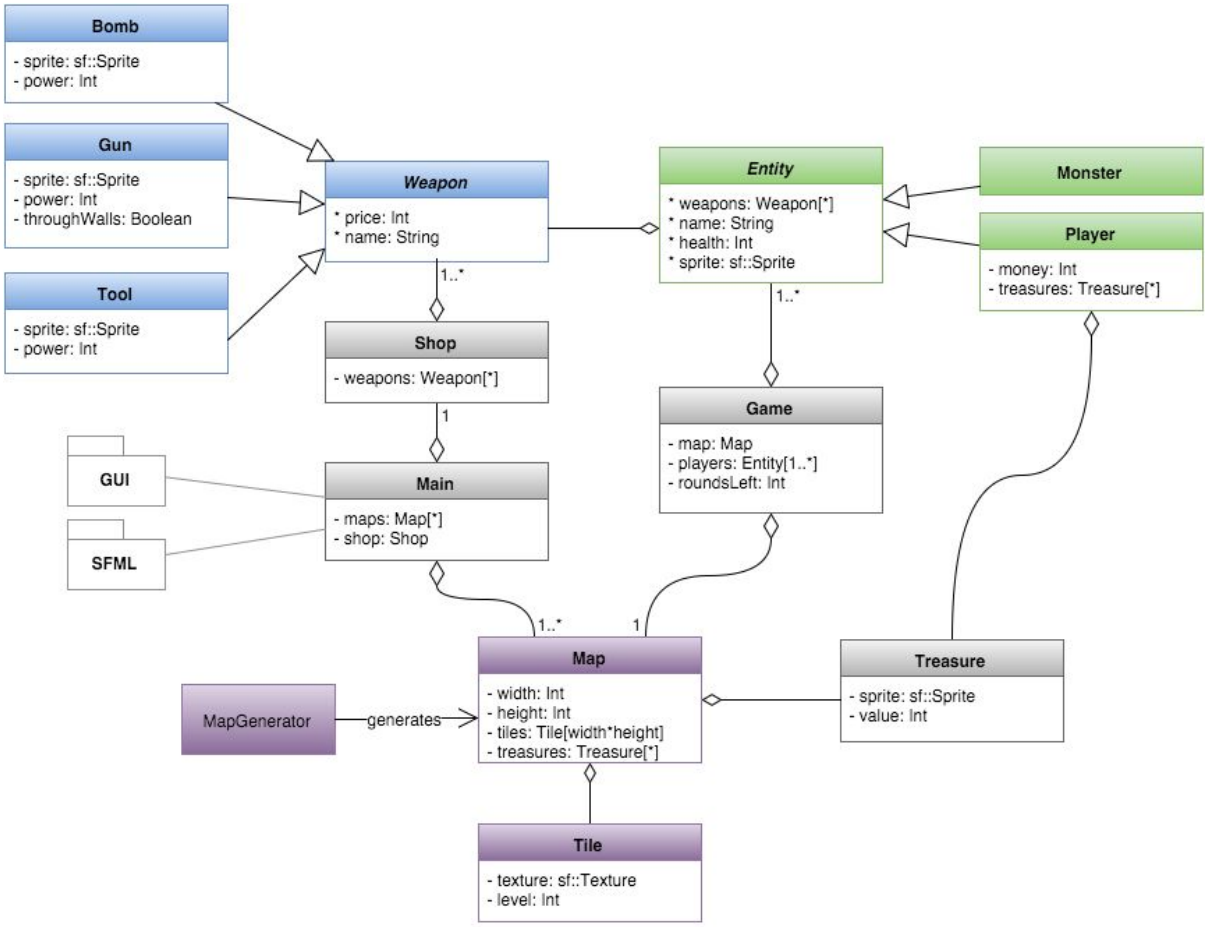
- Level editor
- Netplay
- More weapons
- More themes
- Whatever we come up with during the project

Major Architectural Decisions

We are going to use the SFML library for all visual and audio content. As the IDE we have selected Xcode as all the team members are using Macs. A Makefile will be provided so that Xcode is not strictly necessary for compiling. Rough application structure is defined in the UML class diagram below.

As we are considering netplay as an extra feature, we should take it into consideration from the start to make the possible integration of the feature possible. At the moment we are considering a setup where two or more players could play over the network with one of them acting as the host.

Initial UML Class Diagram



Algorithms

An algorithm is needed for generating random maps. We can use an algorithm that first adds random chunks of the hardest material, next adding the second hardest etc., and finally filling in the map with sand. Treasures are then added to the map at arbitrary locations randomly using rarity weights, overriding the tile with the treasure. Treasures must not be placed on top of the player.

Different algorithms must also be designed for weapons and tools. Weapons, such as different bombs and guns will require different kinds of algorithms for their projectiles and their area of effect, as well as their effect on hitting a target.

Schedule

We have reserved two weeks for implementation of the primary features of the game. We haven’t estimated the time it takes to implement specific features, as it is highly likely that these estimates wouldn’t be realistic.

Week	Activity
45	Project planning
46	Technical plan, project skeleton and initial features
47–48	Implementation
49	Testing and bug fixing, add extra features
50	Testing, final touches

User Interface