Tower Defense Group 4

Author(s): Olavi Kiuru, Kun Ren, Veeti Jaakkola,
Thi Tran provided the structure of the plan but had to drop the course due to personal reasons.

Description

A tower defence game is a simple game where players are faced against waves of enemies trying to reach some goal, usually the player's stronghold or simply the end of the map. The player can build “towers” alongside the path the enemies are taking, using some form of currency. These towers would then hinder the enemies progression by either destroying them or effecting them in other ways, such as slowing them down.

The following is a plan formed to implement a bare bones example of such a game with some added features in order to make the act of playing the game more enjoyable. We will be using the SFML library https://www.sfml-dev.org/index.php to implement our game.

Requirements

Basic Requirements

These are the features of a bare bones tower defence game. At the very least, we will implement these features.

- **Basic features:**
    1. Enemies that follow the given path.
    2. Towers that shoot the enemies within their range.
    3. One life is lost every time an enemy reaches the end of the path. The game is lost if the player loses all their lives. Every level has a certain amount of lives.
    4. Killing enemies gives the player gold, which can be used to buy new towers.
    5. Pause screen between waves of enemies, where the player can build new towers and choose when the next wave of enemies starts.
- **Three different types of enemies:**
    1. A basic tower, shoots enemies within its range
    2. A slowing tower, slows down enemies inside its range
    3. A bomb tower, shoot a bomb when enemies in range, can kill multiple enemies
- **Three different types of towers:**
    1. An enemy that’s killed immediately when it’s hit
    2. An enemy that takes multiple hits to kill
    3. An enemy that splits into multiple kind-1 after killed
- **Five different levels with increasing difficulty**
- **Game controls using the mouse. This includes building towers.**
- **Simple user interface that shows information such as resources, number of waves/enemies etc.**

Additional Requirements

In addition to having a bare bones tower defence game we will also attempt to implement the following features:

- **Tower will be able to be upgraded, this would increase their damage or range, possibly adding different attacks.** This could be done by making the damage variables of a tower an array, which uses the upgrade index to change the value.
- **More different kinds of enemies and towers. For example flying enemies that do not follow the given path. The flying enemies could have their own spawn location that differs from the rest of the enemies.** This can be done by adding different sibling classes, which inherit a common parent enemy class, and giving them a path variable. And different speed/health variables.
- **Stages will have a highscore adding replayability to the game.** Points will be given for example for a fast time and/or having more money and lives at the end of the stage. We could even implement a tower that only increases the score.
- **A capability to alter the defence while enemies are approaching. This could include upgrading and selling towers. Selling a tower returns 50 % of its value. If time allows, an additional feature of moving towers could be implemented. Allowing players to alter the game board during a round makes for a more immersive gameplay experience.** This can be done by not restricting the player's mouse inputs during waves.
- **Immersive audio experience.** This should be doable with basic SFML features.
- **Tower projectiles have different types, e.g. physical and elemental. Some enemies might have resistance to certain projectiles. This adds another strategic element to the game as the player is forced to diversify their tower selection.** We can implement this using a class for the projectiles of towers. Projectiles will have their own types/elements which deal different damage to enemies.
- **Some enemies will take a different path during some of the stages.** Having several "lanes" on a stage adds difficulty to the game since not all towers can shoot at all enemies.
- **A fast forward feature that allows the player to speed up the gameplay.** This could be implemented by adding a multiplier to all movement speeds of enemies and projectiles.

Technical Design

This is the core of the document, and most time is used to research and plan technical solutions prior jumping to implementation. This section defines the technical structure of the project.

Architecture

We will be using a git repository to store the project. And develop the game on visual studio code using SFML library to handle many of the game's features. The SFML library itself includes many useful features that should provide enough to go on to build a functioning tower defence game.

Git Control version

https://version.aalto.fi/gitlab/kiuruo1/cpp-course-project-repository-tower-defense-group-4

Library: Using SFML library: https://www.sfml-dev.org/index.php

Classes Structure

Main Game class:

Runs the core of the game.

Handles the window opening

Handles the time functions and the speed of the game

Opens to the start/level select screen

Clicking on levels will change the view into a level view(seen below)

Initially some of the levels might be locked for the player if they aren't completed.

Initially this will be worked on as a group. In order to create a base for everyone to work on.

Level/GUI, Veeti

Uses: Utils and Map

Each level has a path for enemies.

Multiple waves.

Waves will have different enemies at different phases. Early stages will have simple enemies, while later stages will have more complex enemies.

Waves can be implemented as a vector of vectors of enemies. Where a single wave has a set "schedule" for enemies showing up. During a wave this vector will be iterated and the correlating enemies are spawned on the stage.

Highscore.

A score will be given at the end of the stage, the highest score will be saved.

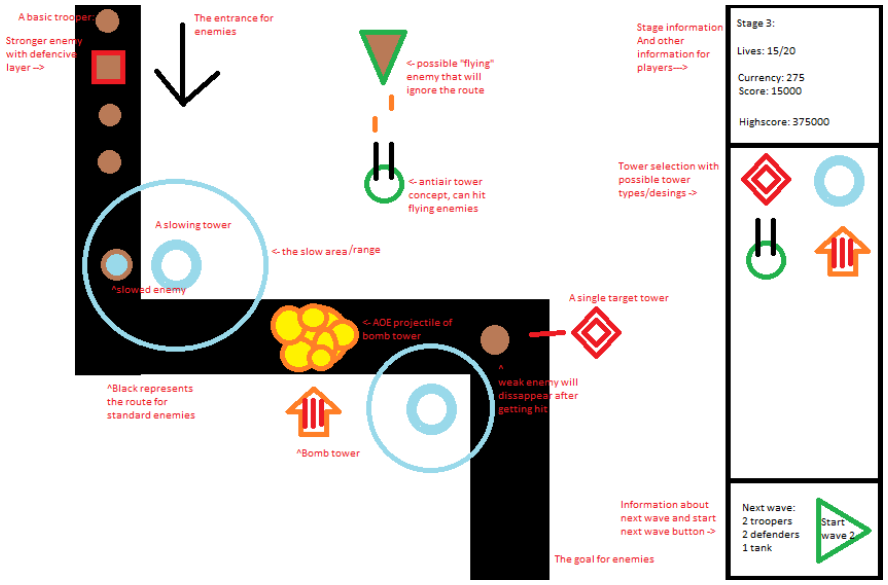
Interface with money, score, wave number and information about wave.

Information could be the number and type of enemies that appear

Interface with towers, showing price, dps and range

Possible locked towers? For example maybe one stage only allows aoe towers.

A possible view for the player with added explanations in red



Once we have a simple system setup for testing, level design and editing will be mostly Veeti's responsibility

Map

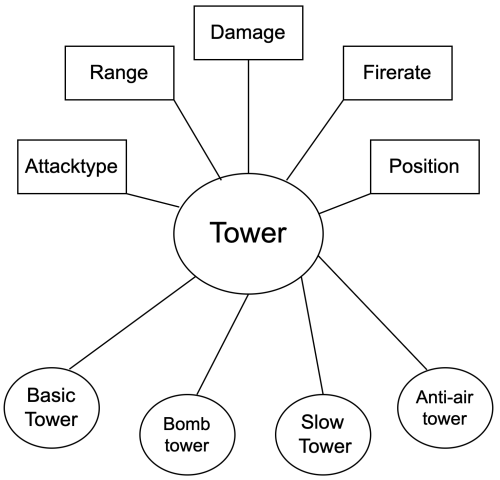
Map should have space for towers and a path/paths for enemies to follow.

Utils

Mouse movement, selection etc.

Reporting for resources, number of waves/ enemies etc.
Utilities will be worked on as a group due to their general usage nature.

Towers, Olavi



Main tower class

Properties:

1. Damage (upgradable)
2. Firerate (upgradable)
3. Range (upgradable)
4. Projectile (Attacktype: (single, aoe, slow, elemental?))
5. Level (determines above)
6. Position (player decides)

Basic attacking tower: will attack a single enemy in range
Bomb tower: will attack close to an enemy damaging other enemies in the blast zone
Slow tower: will slow down all enemies in it's range
Anti-air tower: can hit flying enemies
Towers will have a more intricate design to differentiate them from the simple shapes of enemies.

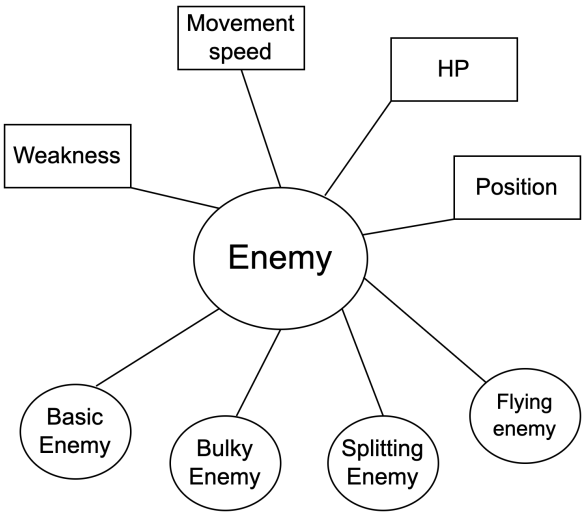
Projectile subclass

Properties:

1. Damage value
2. Damage type/element
3. Projectile speed. This is presumably pretty high, with the exception of aoe(area of effect) damage, where the projectiles can be slower so they can possibly not hit exactly where the target is but still hit it thanks to aoe.
4. Aoe range/single target

Olavi will take charge of the function of towers and their projectiles.

Enemies, Kun



Main enemy class

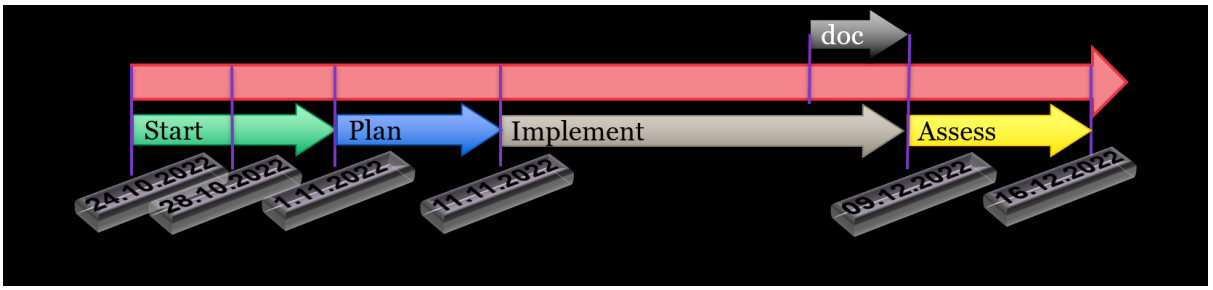
Properties:

1. HP
2. Movement speed?
3. Weakness?
4. Position

Basic enemy: Dies in 1 hit
Bulky enemy: Slower enemy that takes multiple hits to kill
Splitting enemy: Splits to 3 basic enemies when killed
Flying enemy: Immune to certain towers, e.g. bomb tower and does not move along the regular enemy path.
Graphically enemies will have mostly the same colour to indicate to the player which moving objects are enemies. The type difference of enemies will be indicated by their shape. Enemies will have simple shapes.

Ren Kun will be responsible for creation of enemy types.

Milestones



Week 1, 07-11.11.2022: Goal: Finish the plan.

1. Finished plan. Deadline 11.11.2022.
2. Weekly meeting on Friday 11.11.2022 at 14.00

Week 2, 14-18.11.2022. Goal: Get a functioning basic version of the game.

3. Create a simple stage screen for testing.
4. Create initial path, a simple enemy, a single target tower, and the GUI
5. Add main game logic(such as game over when lives end)
6. Create classes and other instances for enemies and towers.
7. Weekly meeting on Friday 18.11.2022 at 14.00.

Week 3, 21-25.11.2022. Goal: Finish all the basic requirements of the game.

8. Create a currency system.
9. Create a wave system and “Start next wave” button.
10. Add the other required tower types
11. Add the other required enemy types
12. Create a level select.
13. Create different levels
14. Weekly meeting on Friday 25.11.2022 at 16.00.

Week 4, 28.11-02.12.2022. Goal: Implement all the extra features. Demo should be ready at the end of the week. Note: Veeti is not present this week due to military service.

15. Add the upgrade system, upgrade tower
16. Add branching paths, one path was split into several paths
17. Implement extra enemy and tower types
18. Add scoring system
19. Add fast forward
20. Weekly meeting on Friday 02.12.2022 at 14.00.

Week 5, 05-09.12.2022. Goal: Submit the polished version of the game.

21. Add sound effects
22. Implement a Makefile for compiling the game.
23. Push the final version to git. Deadline 09.12.2022.
24. Weekly meeting on Friday 09.12.2022 at 14.00.

Week 6, 12-16.12.2022.

25. Peer evaluation Deadline 16.12.2022

References

An example of tower defence game from YouTube:
https://www.youtube.com/watch?v=kSyqhpq9W7w

Planning Notes & Questions & TODO

Availability:

- Veeti: weekdays except monday mornings
- Olavi, Thi: weekdays after 5.
- Kun: Except Wed or Tues afternoon, otherwise free.

Meetings:

- 19:00, 8.11.2022 First meeting
- 17:00, 9.11.2022 Continue working on plan
- 17:00, 10.11.2022 Continue working on plan
- 21:00, 10.11.2022 —---------- | | —----------
- 13:00, 11.11.2022 Finishing touches on the plan and the git commit for the plan
- 13:00, 18.11.2022 Updating others on individual progress, discussion of faced problems and possible solutions.
- 13:00, 25.11.2022 Updating others on individual progress, discussion of faced problems and possible solutions.
- 13:00, 2.12.2022 Updating others on individual progress, discussion of faced problems and possible solutions.
- 13:00, 9.12.2022 Finishing a polishing of the game as a group.