# Core:

Handling of creating and destroying monsters, towers etc.

This will basicly be our main().

# Helper:

**Dist(pos1, pos2)** // Calculates the distance between two positions

**moveTo(curPos,nextPos)** // moves to new position

# Monsters:

**createMonster(name, amount, sprite)** // name is a named enum and sprite refers to sprite file.

# Towers:

**createTower(name, sprite)** // name is a named enum and sprite refers to sprite file.

# Maps:

**createMap(pathname, playerSize)** //filepath of mapfile and amount of players

# Network

Consists of all the packets we’re going to @hook. Consists of incoming and outgoing packets.

Current packets I’m thinking we’ll need:  
 newTick

update

click //perhaps the same as update?

Texture

# Begrundelse for opsætning.

## Core:

Det vi ønsker at noget, som kan skalere uden mere arbejde. Dette kan nemt gøres ved at gøre de enkelte metoder uafhængige af hinanden. Derfor indeholder core ikke andet end oprettelse og destruktion samt noget brugergrænseflade i form af menu.

## Helper:

Klassen indeholder ting, som vi hele tiden skal bruge, men som vi helst skulle være i stand til at holde styr på og ændre i. Ved at rykke disse 2 metoder ind i helper gør vi det hele mere overskueligt og vedligeholdelsesvenligt for os selv.

## Monsters/Towers:

Disse to mapper vil formentlig komme til at indholde 2 filer hver. En sprite- og en monster/tower-fil.  
Sprite filen vil jeg ikke gå ind på, da det bare vil være et sheet af sprites, som man kan referere til. Monster/tower filerne derimod er lidt mere interessante og der er her flere muligheder for at gøre det samme. Enums kan navngives således man ikke referere til et tal, men navnet som det tal det hører til. Jeg tænker at dette vil gøre oprettelsen af monstre meget nemmere, specielt når metoden eksporteres og benyttes andetsteds (så behøver man ikke slå tallet op hele tiden). Derudover vil de enkelte navne henvise til en allerede opsat type (f.eks RAT). RAT vil så nedarve en interface, som vi i fællesskab laver – jeg har lavet et eksempel under /Monsters/monster.common.d og /Monsters/monsters.ts.

## Maps:

Maps skal kaldes fra core og vælges ud fra bruger input. Derfor har den kun navnet på mappet og mængden af spillere. Det eneste denne metode skal bruges til er at sætte nogen faste variabler (hvilke monstre skal spawne på hvilke waves etc) samt loade textures og path. Hvis vi laver vores map-integration på denne måde så kan vi nemt tilføje flere ved blot at tilføje ’map-filen’ og så finde den ved metoden.

## Network:

For at kunne kommunikere mellem bruger og server (hjemmesiden) bliver vi nødt til at sende data frem og tilbage. Denne data samles i det der kaldes packets. Vi kan navngive forskellige grupperinger af data. De 4 nævnte er, hvad jeg forventer vi skal sende, men hvad præcist de hver i sær indeholder må vi diskutere fælles.