

# TDP005 Projekt: Objektorienterat system

## Designspecifikation

Författare

Elliot Johansson, [elljo130@student.liu.se](mailto:elljo130@student.liu.se)

Lukas Freyland, [lukfr510@student.liu.se](mailto:lukfr510@student.liu.se)

Nadim Lakrouz, [nadla777@student.liu.se](mailto:nadla777@student.liu.se)

# 1 Revisionshistorik

Ver.	Revisionsbeskrivning	Datum
1.0	Första utkast	2022-11-25

## 2 Player

Syftet med Playerklassen är att representera den karaktär spelaren styr. Spelaren styr vilka formler player-karaktären använder. Spelaren kommer inte förflytta sig på planen. Vissa formler kommer behöva riktas för att träffa fienderna medan andra inte behöver det. Funktionen `regenerate_mana` kommer sakta återställa manapoäng med `sf::Clock`.

### 2.1

- `int hp` - privat variabel som tilldelas ett heltal.
- `double mana` - privat variabel som tilldelas ett flyttal.
- `sf::Clock clock` - sfml variabel för att hantera tiden.
- `int set_hp()` - ändrar privata variabeln `hp`.
- `int get_hp()` - hämtar privata variabeln `hp`.
- `int get_mana()` - hämtar privata variabeln `mana`.
- `void regenerate_mana()` - ökar privata variabeln `mana` över tid.

## 3 Enemies

"Enemies" klassens syfte är att representera fienderna som i intervaller kommer förflytta sig mot spelaren på spelplanen. Enemies har en `check_collision` funktion som anropas när fiender och spelare kolliderar.

### 3.1

- `float yValue` - förvarar y position.
- `float xValue` - förvarar x position.
- `float size` - förvarar storlek på fiende.
- `double x_movement` - förvarar förflyttningen i x led.
- `CircleShape circle` - förvarar cirkeln som representerar fienden.
- `Vector2f location` - förvarar fiendens position.
- `Texture texture` - förvarar fiendens textur.
- `void update()` - ändrar fiendens position.
- `void render` - ritar ut fienden.
- `CircleShape getCircle()` - returnerar cirkeln som representerar fienden.
- `void setLocation()` - bestämmer fiendens nästa position.
- `Vector2f getLocation()` - returnerar fiendens position.

- `int getDistanceCircles()` - returnerar avståndet mellan två fiender.
- `bool checkCollision()` - returnerar true om två fiender överlappar.
- `double get_x_Movement()` - returnerar hur mycket en fiende ska förflyttas i x led varje förflyttning.

## 4 enemyHandler

- `vector<Enemy*> enemy_container` - vector av enemy pekare.
- `Time render_time` - förvarar frekvensen som fiender ska ritas ut med.
- `Time enemy_create_time` - förvarar frekvensen som fiender skapas med.
- `Clock clock` - skapar en klocka som räknar tiden.
- `void createNewEnemy()` - skapar ny fiende och förvarar i vector av pekare.
- `vector<Enemy*> getEnemies()` - returnerar vector av enemy pekare.
- `void rendering()` - ritar ut varje fiende.

## 5 Spells\_Handler

- `pair<string, int> mana_per_spell` - lista med hur mycket mana varje magiska formel kostar.
- `void handle_input()` - hanterar spelarens inmätning.
- `void handle_combination()` - hanterar kombinationer av magiska formler.
- `void handle_damage()` - beräknar och hanterar damage.
- `void control_spell()` - navigerar spells med hjälp av mus.

## 6

- 

## 7 Engine

- `int current_state` - förvarar vilket state spelet befinner sig i.
- `RenderWindow window` - fönstret som allt renderas i.
- `map<int, unique_ptr> states` - förvarar de olika statesen.
- `Engine()` - konstruktor.
- `int run()` - kör spelet.

## 8 Base\_State

- `unsigned int curr_state` - nuvarande state.
- `virtual Base_State()` - destruktör.
- `virtual void update()` - uppdatera state logiken.

- `virtual void render()` - renderar det som ska visas.
- `virtual int get_current_state()` - returnera nuvarande state.
- `virtual int get_next_state()` - returnerar nästa state.
- 
- 
- 
- 

## 9 utility

- `bool debounce()` - kollar om knapp är nedtryckt.
- `debug_print()` - skriver ut debug text.

## 10 Button\_Manager

- `Button_Manager()` - konstruktor
- `void update()` -
- `void render()` -

## 11 Diagram

