LINKÖPINGS UNIVERSITET

TDP005 Projekt: Objektorienterat system

Designspecifikation

Författare

Mohammad Omar, mohom373@student.liu.se

INSTITUTIONEN FÖR DATAVETENSKAP

Höstterminen 2019
Version 1.1

2019-11-29

Revisionshistorik

Ver.	Revisionsbeskrivning	Datum
0.1	Skapade första utkast av designspecifikation	191129

1 Detaljbeskrivning

1.1 Player

Player-klassen ser till att spelaren har ett spelarobjekt som hen kan styra. Spelarobjektet tar form av en bräda och styrningen sker endast i horizontell led med tangentbordsinmatning. Syftet med spelarobjektet är att försöka stoppa ett Ball-objekt från att träffa DeadZone som är en klass som ser bland annat till spelaren förlorar liv. När spelarobjektet når höger eller vänster vägg uppstår en kollision och brädan hindras från att längre kunna förflytta sig i samma riktning som väggen den kolliderar med.

Player-klassen ärver av Movable-klassen som i sin tur ärver av Entity-klassen.

Dessa variabler och metoder ärver Player-klassen av Entity-klassen:

- x_pos : int - En variabel som anger objektets nurvarande x position.
- y_pos : int - En variabel som anger objektets nurvarande y position.
- width : int - En variabel som anger brädans nuvarande bredd.
- height : int - En variabel som anger brädans nuvarande höjd.
- texture : Texture - Hanterar SFML texture.
- collides() : bool - En funktion som returnerar true eller false beroende på ifall ett objekt kolliderar med ett annat.
- update() : void - En funktion som updaterar variabler.

Dessa variabler ärver Player-klassen av Movable-klassen:

- x_speed : float - En variabel som bestämmer hastigheten av förflyttning i x-led
- y_speed : float - En variabel som bestämmer hastigheten av förflyttning i y-led

Player-klassens egna funktion:

- move() : void - En funktion som förflyttar spelarobjektet.

1.2 Ball

Syftet med Ball-klassen, vilket i det här fallet är en projektil som tar formen av en kvadrat, är att samla poäng genom att träffa poängzonerna som finns på spelplanen. Ball-klassen har också som ett syfte att vara ett hinder för spelaren då hen förlorar ett liv när ett Ball-objekt kolliderar med en Deadzone. Bollen styrs inte av spelaren. Dess riktning bestäms av infallsvinkeln vid kollision med olika objekt.

Ball-klassen ärver av Movable-klassen som i sin tur ärver av Entity-klassen.

Dessa variabler och metoder ärver Ball-klassen av Entity-klassen:

- x_pos : int - En variabel som anger objektets nurvarande x position.
- y_pos : int - En variabel som anger objektets nurvarande y position.
- width : int - En variabel som anger brädans nuvarande bredd.
- height : int - En variabel som anger brädans nuvarande höjd.
- texture : Texture - Hanterar SFML texture.
- collides() : bool - En funktion som returnerar true eller false beroende på ifall ett objekt kolliderar med ett annat.
- update() : void - En funktion som updaterar variabler.

Dessa variabler ärver Ball-klassen av Movable-klassen:

- x_speed : float - En variabel som bestämmer hastigheten av förflyttning i x-led
- y_speed : float - En variabel som bestämmer hastigheten av förflyttning i y-led

Ball-klassens egna funktion:

- trajectory() : void - En funktion som räknar ut projektilbanan och förflyttar bollobjektet.

2 Diskussion av designen

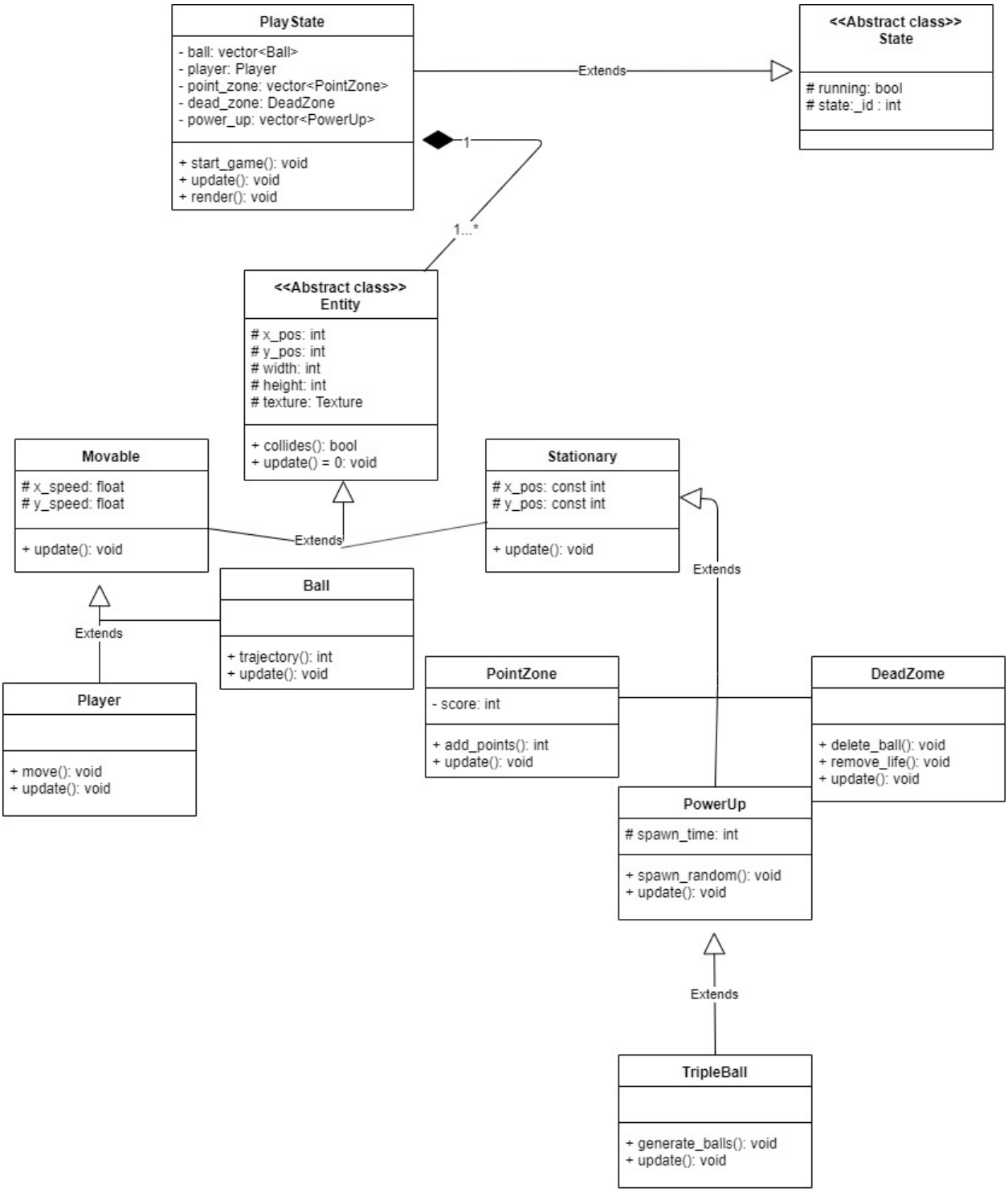
I det här stadiet är designen inte komplett och det är en stor del som kan komma till att förändras under projektets gång.

Nuvarande klassstruktur ser till att det är enkelt att utveckla vidare med nya klasser. Klassen PowerUp som endast innehåller TripleBall för tillfället, kan exempelvis utvecklas med en ny power-up utan större problem genom att låta den nya klassen ärva av PowerUp klassen.

3 Externa filformat

Det externa filformatet som kommer användas för inläsning av spelplan är en textfil(.txt). Innehållet av textfilen är dock inte bestämt vid det här stadiet.

4 Klassdiagram



Figur 1: UML - klassdiagram