**Projekt: Rabbit Wars**

**Version 1.0**

**Skapare: Daniel Bark**

**Spelets mål**

I Rabbit wars tävlar 2 kaniner mot varandra och den som lyckas ha ihjäl den andra kaninen vinner ronden. Efter 3 ronder så vinner kaninen med flest vinster.

**Kaniner**

2 kaniner finns i spelet och styrs av var sin spelare. Varje kanin har 3 liv och kan röra sig åt höger och vänster samt hoppa. Kaniner kan även kasta ut projektiler i syftet att skada sin motståndare.

**Världar**

Kampen tar plats på 3 olika lokationer:

* Gröna Skogen
* Mörka Grottan
* Rymdstationen

I varje värld så använder kaninerna olika förmål att kasta på varandra för att orsaka skada.

I skogen används morötter, i grottan fladdermöss och i rymden raketer. Gravitationen skiljer sig mellan dem olika

**Projektiler**

I Rabbit wars finns 3 olika projektiler:

* Morötter – kastas snett uppåt i en båge och kan landa på motståndaren ovanifrån.
* Fladdermöss – kastas rakt fram och flyger i en rak bana framåt.
* Raketer – Flyger rakt fram men ändrar då och då riktning slumpmässigt.

En projektil som träffar en kanin gör att kaninen förlorar ett av sina 3 hjärtan.

Alla projektiler kan uppgraderas till superprojektiler om kaninen inte kastar ut någon projektil på en liten stund. En superprojektil gör dubbel skada jämfört med en vanlig.

**Ikoner**

Ikoner finns för att visa antalet liv samt om kaninen har en projektil eller superprojektil att kasta för tillfället.

**Mover**

Moverklassen innehåller ett gäng olika metoder som till exempel gravitation som Actors kan ärva för att få önskad rörelsemönster. Kaniner och morötter använder till exempel gravitation medan fladdermöss inte behöver gravitationen så dem motverkar den med sina vingar.

**Måltavlor**

Slumpmässigt placeras det ut måltavlor i världen som kaninerna kan skjuta med sina projektiler. Om en kanin träffar en måltavla blir kaninen osårbar för en stund.

**Att bygga vidare på spelet**

Att bygga nya banor till spelet bör vara ganska smidigt. Man kan ärva alla object förutom projektilerna då dem är specifika per värld. Man måste implementera nextlevel metoden då den är abstract. Det man måste komma ihåg är att ändra i den befintliga sista världens nextLevel metod att den ska skapa upp den nya världen och inte skapa upp EndScreen.

Att variera gravitationen i världen görs på varje enskild Actor. Gravitymetoden ligger i Moverklassen.

För att designa en ny projectil kan man ärva funktionalitet från projektilklassen och rörelse från mover. Sedan kan man implementera lite eget i den egna klassen för att särsklija projektilen från dem andra projektilerna.. Se till exempel RandomTurn metoden i Rocket klassen.

**Tankar kring lösningen**

På grund av att jag lärde mig tänket kring objektorientering under tiden jag skapade detta spelet så blev det inte riktigt så bra som jag ville ur ett objektorienteringsperspektiv. Att man tar beslut om spelets design under tiden man skriver koden försvårar också att skriva snygg generell kod man kan återanvända. Om man hade skrivit om spelet från grunden nu när man vet hur slutprodukten ser ut skulle man kunna göra en mycket snyggare lösning med avsevärt färre rader kod. Jag hade till exempel velat ha en klass för Rabbit1 och Rabbit2 men kände att det skulle ta för mycket tid att göra det nu när lösningen är färdig. Jag hade även velat baka ihop RabbitWorld1 2 och 3 till en class där man med inparametrar till konstruktorn skickade med vilken typ av projektil som skulle användas i världen.

**Tankar kring spelupplevelsen**

Spelmässigt är jag mycket nöjd. Jag tycker att spelet är kul och kompetitivt och jag har fått mycket positiv feedback från dem som provspelat.