

Planetsimulator

Johannes Olegård

Anton Olsson

25 april 2013

1 Programbeskrivning

Vi ska skriva ett program som simulerar ett litet universum från ett 2D-perspektiv. Användaren kan skapa planeter i olika storlekar, massor, färger, hastigheter, accelerationer, etc. och planeter kommer sedan interagera med varandra (t.ex. skapa omloppsbanor).

2 Användarbeskrivning

Applikationens användargrupp är seende mellan 13-90 med åtminstone lite datorvana som är intresserade av att experimentera med fysik. Användaren förutsätts ha viss kunskap i relevanta fysikaliska begrepp (t.ex. vet ungefär vad massa och radie är) då programmet ej förklarar dessa. Vidare förutsätts användaren förstå engelska då text i programmet är på engelska.

Programmet skulle t.ex. kunna användas av högstadiesbarn för att lära sig experimentera med fysik för att förstå hur gravitation, etc. påverkas av de olika storheterna.

3 Användarscenarier

3.1 Scenario 1

Användaren är en datalog på kth som går första året. Fysikkunskaperna är goda. Användaren vill simulera ett solsystem med 3 planeter i omloppsbanor runt en stjärna.

Programmet startas och ett fönster öppnas i mitten av skärmen med tomt solsystem och en meny kant till vänster där det finns parametrar som t.ex. massa och radie att ställa in. Användaren ställer in stjärnans egenskaper och klickar sedan i mitten av skärmen där det då skapas en ifylld cirkel.

Användaren ställer in nästa planet och håller inne mus knappen där planeten ska skapas och drar sedan pekaren åt det hållet initial hastighet ska vara. Medans användaren drar med musen så syns en pil från planeten till pekaren med text brevid om hur hög hastigheten blir när användaren väl släpper.

En streckad ellips visar hur omloppsbanan kommer bli. När användaren släpper så börjar den nyskapade planeten röra sig i en ellips runt stjärnan. Användaren repeterar mönstret för 2-3 planeter men märker denna gång att när hen håller på att placera (musknapp nertryckt och vektorn för hastighet syns) en ny planet så pausas simulationen för att underlätta placeringen.

3.2 Scenario 2

Användaren är en idiot. Fysikkunskaperna är dåliga.

Användaren klickar ut många planter, solar, etc. Dessa börjar kollidera och flyga utanför skärmen. Användaren tröttnar snart.

4 Testplan

Vi testar programmet framförallt genom användartestning.

Användartestningen ska ske genom att ta tag i lämplig person i godtycklig kth korridor och få denne att utföra en eller flera av följande uppgifter:

1. Skapa ett solsystem med en planet och en tillhörande måne.
2. Justera omloppsbanan på en yttre planet i ett existerande solsystem med 2 orbitaler så att planeten hamnar innanför den andra.
3. (svår?) Skapa ett dubbelstjärnigt solsystem med en tillhörande planet.
4. (svår) Skapa en lite galax. Många stjärnor i mitten och fler hela solsystem som kretsar runt dessa.

Innan användaren börjar ska vi kort förklara med ord (utan att visa) vad vi vill att hen ska försöka utföra. Vi ska också tydligt förklara att eventuella misslyckanden beror på brister i det grafiska gränssnittet och inte användarens bristande intelligens (även om det säker bidrar). Vi behöver tydligt anmärka att programmet är under utveckling och därför är det fullt möjligt att buggar gör uppgiften omöjlig. Det finns otydligheter i det grafiska gränssnittet och vi ber därför användaren att säga högt vad hen tänker så vi kan förstå vad som behöver ändras/förtydligas för en användarvänlig design.

5 Programdesign

Programmet ska använda opengl för det grafiska med hjälp av biblioteket libgdx¹. Artemis entity system² kommer användas för att lättare separera koduppgifter i mindre klasser där varje klass gör en (1) enskild sak.

Artemis använder sig av komponenter och system, där komponenter håller data och systemen bearbetar dom. Ett system använder endast några få komponenter och arbetar endast på objekt som har dom komponenterna.

Planeterna ska ha följande komponenter:

- Position
- Hastighet
- Acceleration
- Massa
- Storlek

¹<http://libgdx.badlogicgames.com/features.html>

²<http://gamadu.com/artemis/index.html>

- Färg

De system som behövs är:

Gravitation behöver använda *position*, *massa* och *acceleration* för att räkna ut gravitations krafter mellan planeterna.

Acceleration behöver *acceleration* och *hastighet* för att förändra hastigheten.

Hastighet behöver *position* och *hastighet* för att uppdatera positionen.

Ritande behöver *position* och *storlek* för att rita planeterna på skärmen.

6 Tekniska frågor

1. Vilken algoritm ska användas för att justera planets position, hastighet och acceleration? Ska t.ex. alla planeter jämföras med alla andra planeter i universumet, eller ska algoritmen vara selektiv och inte räkna planeter för långt borta?
2. Hur hanteras kollisioner? Studsar, spricker eller smälts planeter samman? Hur kommer kollisionssystemet påverka resten av programmet? Behövs systemet snabbas upp?
3. Behöver programmet snabbas upp med parallellisering? I så fall, hur? Bör läsning av delade variabler undvikas?

7 Arbetsplan

Arbetet pågår under vecka 17 t.o.m. 20. Under denna tid är det 3 övningstillfällen: fre 26 april (v17), fre 3 maj (v18) och ons 15 maj (v20).

Projektplanen har deadline vid den första övningen. Slutrapporten och programmet har deadline vid den sista övningen. Vid varje övningstillfälle ges en muntlig lägesrapport.

Vecka 17 spenderas på att skapa ett git-repository med ett kodskelett och att skriva projektplanen. Om det finns tid över läggs den på att till viss del att börja fylla kodskelettet.

Vecka 18 ska användas till att skriva så mycket av programmet som möjligt. Om det finns tid över börjar användartestningarna v18 och fullgörs annars (senast) v19.

Vi räknar med ett fullt fungerande, debuggat och användartestat programmet i mitten av v19, 1 maj. Slutrapporten ska skrivas så mycket som möjligt parallellt med resten av arbetet, men framförallt slutet av vecka 19 används till rapporten. Om det finns tid över kan det användas till debugging i programmet.

Vi räknar med att slutrapporten är klar i slutet av vecka 19, fre 11 maj.

Vecka 20 (totalt 2 dagar innan inlämning) används endast om något moment tog längre tid än förväntat.