Projektarbejde, Snake

Anna lgaard Nielsen - s144437
Christian Sholm Andersen - s103080
Mathias Enggrob Boon - s144484
Van Anh Tri Trinh - s1444449

18. januar, 2015

Contents

1.1 Struktur af Simpel-Snake

Snake-spillet er lavet efter et MVC-design, hvormed selve spillet, styringen af spillet og den visuelle reprsentation af spillet holdes adskilt i tre dele.

1.1.1 Styring

I simpel-snake bruges der kun 4 taster til input, nemlig de fire retningstaster, som derfor er defineret i control. Control-klassen har en metode, keyPressed, der kaldes hver gang der tastes. Hvis tasten er en af de fire retningstaster, flyttes slangen i den tilsvarende retning. Hvis ikke, sker der intet.

1.1.2 Model

Spillets model bestr af en rkke klasser, der tilsammen udgr selve spillet. Klassen "Field" bruges til at definere et objekt, som kan opdele banen. Funktionenen equals er defineret i denne klasse, og bruges til at undersge om to objekter ligger p samme felt, f.eks. slangens hoved og blet. blet er defineret i klassen "Food", hvor datafeltet "position" afgr dets nuvrende position. Slangen selv er defineret i klassen "Snake". Slangens krop bestr af en rkke felter, hvoraf det frste er hovedet, og de resterende er kroppen. Koordinaterne for disse felter er gemt som elementer i en ArrayList kaldet "positions". Det frste element er slangens frste led, hovedet, andet element er slangens andet led osv. Nr et nyt led tilfjes, tilfjes et nyt element til listen. Funktionen createStartingSnake laver slangen ved at bestemme banens centrum, og tilfje to elementer til "positions" med centrumfeltet og feltet til venstre for dette felt.

Slange

ArrayList til at holde slangens punkter. Element 0 er altid hovedet - Sidste element er altid halen Direction = Sidste retning slangen bevgede sig. Nr slangen flytter sig, opdateres dens felter "bagfra", dvs. hale bliver til nstsidste position, nstsidste position bliver tredjesidste osv. Sidst flyttes hovedet til den nye position.

View (Brugerflade)

1.2 Udviklingsproces - Simpel Udgave

1.2.1 Controller

1.2.2 Model

Field vs. point til at bestemme koordinater

Opbygning af banen

Til designet af selve banen, som slangen bevger sig p, forel to muligheder. Den ene var at lave et to-dimensionelt array af datatypen enum, hvis strrelse afgr banens endelige strrelse. Et array [10][5] vil f.eks. give en bane med lngden 10 og bredden 5. Elementerne i arrayet kan da f.eks. vre et blankt felt, et ble, et led af slangen osv. Dette gr det nemt at introducere nye spilelementer i fremtiden, f.eks. bonus-point, vgge, miner osv. Spillet kan da nemt visualiseres ved at definere et billede for hvert spilelement. Programmet kan da tegne det tilsvarende objekt p korrekte plads. ULEMPER - observer?

En anden metode er, at lade de forskellige spilelementer vre defineret i deres egne klasser, s f.eks. SnakeFood er en klasse for sig selv, SnakePlayer er en klasse for sig selv osv. Hver klasse har de funktioner, der er relevante for dem, f.eks. getPosition for at give deres nuvrende position. Programmet tegner da spillet hver gang en tur afsluttes, dvs. nr alle elementer som skal ndres, er ndret. Programmet har fet defineret billedet for de forskellige elementer, og modtager deres position vha. en getPosition metode. Ulempen ved denne metode er, at tegning af programmet gres mere kompliceret. I den frste metode er alle felter allerede defineret, og for at ndre dem behves det blot at ndre vrdien. nskes et spilelement ndret eller introduceret med den anden metode, skal der laves et nyt element.

Den anden metode blev valgt til spillet,

Banens strrelse

Et vigtigt element i spillet er "blet" slangen gr efter. Idet blet har en bestemt position, blev en metode lavet, sledes at blet fr en ny tilfldig position, nr slangens hoved nr blet. HVORDAN?!

Snake

Da slangen i snake-spillet bestr af en rkke felter, som alle har netop en koordinat i forbindelse med de resterende, er en effektiv mde at bestemme slangens position p en LinkedList, idet denne datastrukturer er fleksibel i strrelse og passer til formlet. I frste omgang blev en ArrayList benyttet, men blev erstattet, idet slangens frste led (hovedet), altid placeres som element 0. Det er da mere effektivt at benytte en LinkedList, for at undg at skulle flytte alle elementer, hver gang lngden ges. Bevgelse af slangen fungerer er opdelt vha. en rkke if og

else statements. Det undersges frst, om slangen bevger sig ind i et felt, hvor der er et ble. Hvis dette er sandt, tilfjes der et nyt led til slangens LinkedList p position 0 med samme position som blet. Dette nye element er da slangens nye hoved. Er der ikke et ble i feltet, undersges det om et af slangens andre led (undtagen halen) har samme position. Hvis sandt, slutter spillet, idet slangen har ramt sig selv. Hvis ingen af disse kriterier er opfyldt, flyttes slangen normalt i den nskede retning. Kontrol af retning og bevgelse ud over banen behandles af controller-delen.. Score For at implementere scoren blev der fremlagt to lsninger. Enten at lade scoren vre et datafelt i game-klassen, eller at lade det vre en klasse for sig selv. Ved at lade scoren vre et datafelt, bliver implementationen simplere. At lade scoren vre en klasse for sig selv har derimod fordelen, at der kan tilfjes en observer til Score-klassen, som dermed kun opdateres, nr scoren ndrer sig. Scoren bliver dermed kun gentegnet, nr scoren ndrer sig. Hvis score derimod er et datafelt, tegnes scoren efter hver tur, ogs selvom scoren er undret, hvilket er mindre effektivt. Forskellen er dog minimal, og det blev derfor prioriteret at holde scoren som datafelt, hvormed det ogs bliver simplere at implementere nye score-relaterede funktioner i fremtidige udgaver.

1.2.3 Brugerflade og visualisering af programmet

Tegning af banen

Vinduestrrelse

Omrdet, som spillet foregr p, skal kunne bestemmes til at vre mellem 5x5 og 100x100. Dog er felterne defineret som en brk af vinduets samlede strrelse, dvs. at f.eks. et felt i en 10x10 bane er 1/10 af vinduets strrelse. Dog kan en stor bane krve, at selve vinduet ogs er af en hvis strrelse, s de enkelte felter ikke bliver for sm. F.eks. er det svrt at spille med en banestrrelse p 75x75, hvis vinduets strrelse er 375x375, idet de enkelte felter da kun bliver 5 pixels brede. For at lse dette problem, kunne der vlges mellem to lsninger. Enten kunne felternes strrelse fastlgges, og vinduets strrelse justeres efter dette. nskes det f.eks. at felterne altid har strrelsen 20x20, og at banen skal vre 15x25, vil vinduets strrelse blive 300x500. Fordelen ved denne metode er at det sikres, at banen altid er synlig, og at der ikke opstr problemer, fordi forholdet mellem vinduets strrelse og banen strrelse ikke passer sammen. Ulempen ved metoden er, at store baner kan blive for store til at vre p en normal skrm, f.eks. vil en bane med felter af strrelsen 20x20 og banestrrelsen 75x75 fylde 1500x1500.

Den anden metode, at gre vinduet justerbart, lser dette problem, idet vinduets strrelse frit kan justeres som nsket. Denne metode introducerer dog et andet problem, nemlig at felternes strrelse skal skaleres til at passe vinduet. Nogle oplsninger af vinduet vil ikke vre et multiplum af banens strrelse, hvormed elementerne i spillet vil blive aflange. Dette kan dog lses ved at lave en baggrund og lse banens forhold, hvormed banen altid fylder mest muligt af vinduet ud, og den resterende plads bliver udfyldt af baggrunden. For at give brugeren den bedste mulighed for justering blev metode nummer 2 valgt, hvormed banerne altid er synlige til at starte med, og kan justeres efter nsker derefter.

1.3 Struktur af endelig version

1.3.1 Styring

1.3.2 Model

1.3.3 Brugerflade

1.4 Udviklingsproces af avanceret version

1.4.1 Optegnelse af slange

I simpel-snake kan spilleren se hvor han har vret, men ikke i hvilken rkkeflge. Spilkvaliteten kunne dermed forbedres, ved at lade spilleren se, hvordan slangen har bevget sig. Det nskede resultat ville vre, at hvert af slangens led er forbundet med det forrige og nste, og s har fri plads i de omrder, der ikke er forbundet. For at opn dette blev der lavet en ny klasse, Bodytype, til at definere hvordan slangens led skal se ud. I snake-klassen blev en ny ArrayList tilfjet, nemlig body, som bruges til at bestemme Bodytype for de forskellige led. Bodytype(0) er dermed hovedet, bodytype(1) er frste led osv. Hver bodytype bestr af to datafelter af typen "relation", hvis vrdi er afgjort af det forrige og nste led. Er det forrige led f.eks. over det nuvrende, er prev = ABOVE. Er det nste led til hjre for det nuvrende led, er next = BESIDERIGHT. Leddenes Bodytype skal naturligvis opdateres efter hver tur. Dette gres af funktionen updateBody, som vha. en for-lkke gr gennem alle elementerne. Det undersges da, om positionen for det forrige og nste led er over, under, til hjre eller venstre for det nuvrende led, og prev samt next sttes til at passe dette. updateBody kaldes hver gang slangen bevger sig.

1.4.2 Lyd

Som en mindre tilfjelse til spillet er lyd tilfjet i form af Audio-klassen. Funktionerne i Audio-klassen bruget til individuelt at spille

1.4.3 Hovedmenu

1.4.4 Automatisk bevgelse

1.4.5 Variatoner af baner

1.4.6 Multiplayer