NTNU

Kunnskap for en bedre verden

Øvingsforelesning 6 - Oppgaver

TDT4100 Objektorientert programmering

Onsdag 12.02.2020

Oppgaver

Vi skal utvide implementasjonen vår av en forenklet versjon av spillet Snakebird til å være en grafisk versjon.

Koden som er utdelt er en implementasjon av en tekstbasert versjon av spillet gjort i forrige øvingsforelesning og kollokvium. Koden består av to klasser, **Tile** som representerer en rute på spillbrettet og **Game** som representerer hele brettet og logikken i spillet.

Spillregler

En oversikt over spillets regler finnes i forrige øvingsforelesning.

NTNU

Pakker

Utdelt kode og oppgavene finnes i mappen foreksempel/src/of6.

foreksempel/src/of6.kode

Her finner dere utdelt kode og kan skrive deres egen kode.

foreksempel/src/of6.underveis

Her blir kode lagt inn rett etter felles gjennomgang.

foreksempel/src/of6.tester

Her ligger kode som kan brukes til å teste at en oppgave er løst riktig

foreksempel/src/of6.lf

Blir gjort tilgjengelig i etterkant og innholder et løsningsforslag for oppgavene.

NTNU

Oppgave 1

7 min

Motivasjon

For å kunne vise spillet grafisk trenger vi representasjon av hvordan spillvinduet skal se ut.

Oppgavetekst

Bruk SceneBuilder til å lage en FXML fil som representerer et vindu med et Pane for spillbrettet og fire knapper.

Testing

Se på FXML filen i SceneBuilder.

NTNU

Oppgave 2

5 min

Motivasjon

Vi har nå en representasjon av hvordan vinduet vårt ser ut, men trenger kode som kan vise denne representasjonen.

Oppgavetekst

Lag en tom kontroller klasse og sett opp en app klasse, slik at vi kan kjøre programmet og få opp vinduet vi lagde i forrige oppgave.

Testing

Ved riktig implementasjon skal vinduet som FXMLen fra forrige oppgave representerer vises på skjermen ved kjøring

NTNU

Oppgave 3

5 min

Motivasjon

Før vi kan begynne å tegne spillbrettet må vi ha logikk som oppretter og holder styr på en spill instans.

Oppgavetekst

Implementer en initialiseringsmetode i kontrollerklassen, som oppretter et brett og tar vare på dette brettet som et felt i kontrollerklassen. For brettet som ble brukt i forrige øvingsforelesning se innholdet i filen "brett.txt" i mappen for tester.

Testing

Ikke veldig lett før vi begynner å tegne spillet.

Oppgave 4

7 min

Motivasjon

Vi har nå et fungerende oppsett og kan begynne å tegne brettet vårt. Det første vi må gjøre er å generere et grafikkelement for hver rute i brettet.

Oppgavetekst

Implementer en metode **createBoard** i kontrollerklassen som fyller **Pane**-objektet fra oppgave 1 med et rutenett av **Pane**-objekter med størrelse 20x20. Kall denne metoden i initialiseringsmetoden.

Testing

I og med at ingen av **Pane**-objektene vi lager har noen bakgrunnsfarge, kan det være vanskelig å se at vi har fått laget et ordentlig rutenett. Koden i filen "oppgave4-style.txt" kan brukest for å tegne en sort kant rundt en rute.

Oppgave 5

5 min

Motivasjon

For å tegne brettet trenger vi å vite hva slags farge hver av rutene skal ha.

Oppgavetekst

Lag en funksjon **getTileColor** i kontrollerklassen som tar inn et **Tile**-objekt og returnerer en fargekode som representerer hva slags type rute det er. Du trenger ikke å ta hensyn til at hodet til slangen burde representeres på en annen måte. Filen "oppgave5-farger.txt" inneholder greie fargekoder for representasjonen.

Testing

Kjøring av koden i filen "oppgave5-test.txt" i slutten av initialiseringsmetoden i kontrollerklassen, skal ved riktig implementasjon skrive ut fargekodene i samme rekkefølge som de er gitt i filen "oppgave5-farger.txt".

NTNU

Oppgave 6

7 min

Motivasjon

Vi er nå klare til å fargelegge brettet.

Oppgavetekst

Lag en funksjon **drawBoard** i kontrollerklassen som oppdaterer bakgrunnsfargen til hvert av **Pane**-objektene som representerer rutene i brette med den fargen som **getTileColor** returnerer for den spesifikke ruten. **drawBoard** skal kalles etter **createBoard**.

Testing

Ved riktig implementasjon skal kjøring av appen vise brettet med farger.

Oppgave 7

5 min

Motivasjon

For at spilleren skal kunne skjønne hvor hodet til slangen er, må vi representere dette med en annen farge.

Oppgavetekst

Lag en funksjon **isSnakeHead** i **Game** som tar inn et **Tile**-objekt og sjekker om dette er hodet til slangen. Endre **getTileColor** til å bruke denne funksjonen til å sette en annen farge på hodet til slangen. Fargekoden "#1db121" en mørkere grønnfarge enn den som brukes for resten av slangen.

Testing

Ved riktig implementasjon skal kjøring av appen vise brettet slik som etter forrige oppgave, men nå med en annen farge på hodet til slangen.

NTNU

Oppgave 8

5 min

Motivasjon

Vi har nå en grei grafisk representasjon av brettet og kan begynne å se på bevegelse.

Oppgavetekst

Implementer funksjonene **handleUp**, **handleDown**, **handleLeft**, **handleRight**. Disse skal flytte slangen et hakk i den retningen som metodenavnet tilsier, og oppdatere grafikken.

Testing

Prøv å kalle metodene i initialiseringsmetoden til kontrollerklassen og se at grafikken blir oppdatert riktig.

NTNU

Oppgave 9

5 min

Motivasjon

Det eneste som mangler for å flytte slangen er at trykking på knappene kaller riktig metode.

Oppgavetekst

Endre FXMLen slik at når de fire knappene trykkes på, så kjøres den respektive **handle<direction>** metoden.

Testing

Kjør spillet og se at trykking på knappene flytter slangen.

NTNU

Oppgave 10

10 min

Motivasjon

Vi har nå et fungerende spill, men hvis vi vinner eller taper virker det kun som om spillet slutter å fungere.

Oppgavetekst

Endre **drawBoard** klassen slik at hvis spillet er vunnet kommer det opp en grønn tekst som sier "You Won!" og hvis spillet er tapt kommer det opp en rød tekst som sier "You Lost!".

Testing

Prøv å tape og å vinne og verfiser at meldingen dukker riktig opp.