Semesteroppgave: datastruktur og grafikk

DATS1600 oppgavesett 2.

I dette oppgavesettet skal dere implementere en enkel datastruktur for å representere spillebrettet til GoL. I tillegg, skal dere implementere metoden som tegner spillebrettet til skjermen.

Flerdimensjonale tabeller i Java

Flerdimensjonale tabeller er presentert i forelesning (uke 5).

Spillebrettet i GoL er definert som et 2D rektangulært grid, der celler enten er i live eller er døde. Dere skal i denne oppgaven implementere representasjonen av dette brettet. Vurder og diskuter ulike representasjoner. Jeg vil anbefale en struktur som gjør det mulig å endre representasjonen av brettet senere i utviklingen uten å gjøre store endringer i resten av koden.

I første omgang, burde dere sette verdier i koden direkte (dvs. i deklarasjonen av brettet). Dere skal senere implementere funksjonalitet som angir tilstander til celler ved å lese data fra fil. Eksempel på deklarasjon:

```
private byte[][] board = {
                          { 1, 0, 0, 1 },
                          { 0, 1, 1, 0 },
                          { 0, 1, 1, 0 },
                          { 1, 0, 0, 1 }
                         };
```

Grafikk i JavaFX

Canvas klassen i JavaFX kan brukes til å tegne 2D grafikk til et GUI området. Bli kjent med denne klassen her: http://docs.oracle.com/javase/8/javafx/graphics-tutorial/canvas.htm#JFXGR214

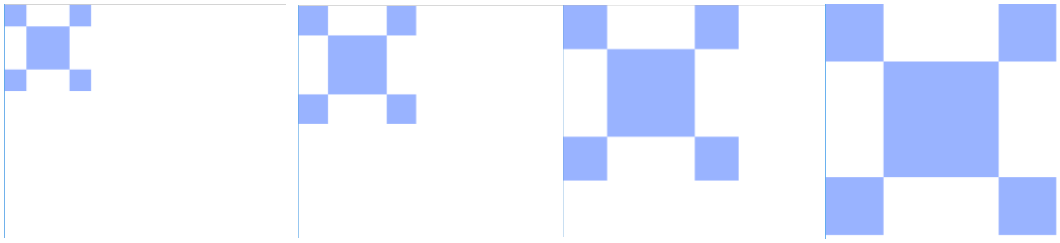
Se FXML_eksempel for en demonstrasjon av Canvas klassen.

Opprett en draw(GraphicsContext gc) metode. Metoden skal tegne spillebrettet til skjermen. For eksempel, kan man bare tegne celler som er i live (ved bruk av en eller flere farger). Døde celler vil da korrespondere til bakgrunnsfargen.

Opprett et objekt av typen til spillet i Controller klassen for GUI programmet. Det er to valg her: opprettelse av objektet ved oppstart av programmet (via initialize(…) metoden), eller opprettelse av objektet via en hendelse (f.eks. klikk på en knapp). Lag en draw() metode relatert til Canvas objektet i Controlleren som bruker draw() metoden dere laget over. Eksempel på resultat:

Grafikken beskrevet over er ikke spesielt 'attraktivt' å se på. Utforsk ulike måter for å forbedre visningen av spillebrettet for å gjøre brukeropplevelsen bedre (bruk fantasi her). Eksempler på grafiske aspekter er visning av grid, bruk av forskjellige farger (f.eks. ved bruk av tilfeldighet eller posisjon), og bruk av effekter. Tips: JavaFX sin ColorPicker kan brukes til å la brukeren velge farger.

En annen fri variabel, i tillegg til farge, er størrelsen på en celle slik den vises på skjermen. Opprett en medlemsvariabel, cellSize, og inkorporer den i draw(…) metoden. Brukeren burde kunne manipulere denne variabelen via GUIet, for eksempel via Slider eller scroll. Eksempel på manipulering av cellSize:



Møte med kontaktperson

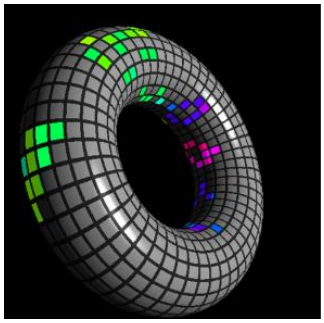
Etter utførelse av oppgavene over skal dere har kommet godt i gang med programmeringen. Dere skal nå avtale et møte med kontaktperson for å hovedsakelig diskutere samarbeidet i gruppen og prosjektutførelsen. Det er deres ansvar å kontakte deres kontaktperson for møte. Samtaleemner:

- Hvordan har dere arbeidet med oppgavene hittil i faget?
- Bidrar alle medlemmene i gruppen likt? Hvis ikke, må samarbeidet endres.
- Hvordan utfører dere prosjektet, iterativ utvikling eller etter vannfallsmetoden? Bruker dere verktøy som Kanban Board og planning game? Kontaktperson kan komme med konstruktive forslag til hvordan prosjektutførelsen kan forbedres.
- Fordeler og ulemper med deres nåværende løsning burde også diskuteres.

Utvidelsesoppgave

NB: det anbefales å implementere kjernen til programmet (sett 5) før dere starter med utvidelsesoppgaver.

For studenter kjent med 3D grafikkprogrammering (eller studenter som ønsker å lære det), kan også spillebrettet programmeres i 3D. For eksempel, kan dere bruke et plan for spillebrettet og bruke lys og shading for å belyse det. Alternativt, kan brettet defineres over 3D objekter, slik som vist i bildet under (implementert med JWJGL).



Det er flere måter å oppnå dette på. De to hovedalternativene er å enten bruke JavaFX sitt grafikkbibliotek eller å bruke et eksternt bibliotek, som for eksempel OpenGL.

For mer informasjon for støtten til JavaFX, se: http://docs.oracle.com/javase/8/javafx/graphics-tutorial/javafx-3d-graphics.htm

En ulempe med JavaFX sitt bibliotek er at det er relativt nytt og dokumentasjonen for denne delen av biblioteket er relativt dårlig. Man har heller ikke tilgang til den underliggende grafikkprosessoren, noe som fjerner en del frihet.

Det primære alternativet er OpenGL, der man i Java bruker et wrapper bibliotek (OpenGL er et C bibliotek). Jeg har hatt best erfaring med JWJGL 3 biblioteket, som ser ut til å være best vedlikeholdt. Merk at OpenGL (og JWJGL) er et prosedyrebasert bibliotek, så vær forberedt med å tenke etter denne stilen fremfor den objekt-orienterte stilen som du kanskje er mer vant til.

Et problem er å kople OpenGL opp mot JavaFX (noe som ikke er anbefalt). Man kan integrere denne delen av programmet med Swing biblioteket, da JWJGL er bedre integrert med Swing. Jeg vil imidlertid anbefale å skrive et separat program eller klasse uten bruk av JavaFX GUI elementer (eventuelt interaksjon via mus og tastatur) som bruker de samme logikk klassene fra JavaFX programmet (det går for eksempel fint an å ha to main-prosedyrer i samme Java prosjekt).

https://www.lwjgl.org/