

I110 Høsten 2001, obligatorisk oppgave 3

Krav til innlevering

Den obligatoriske oppgaven skal leveres skriftlig.

For at denne obligatoriske oppgaven skal bli godkjent må følgende leveres:

- Oversikt

Et dokument som inneholder en oversikt over innleveringen. Den skal inneholde en kort beskrivelse av alle filene som blir levert inn og skal også oppgi navn, gruppe og email-adresse. Hvis man ikke går på noen gruppe bør man skrive navnet til gruppelederen som har rettet den første obligatoriske oppgaven. Dette dokumentet er viktig siden det hjelper den som skal rette oppgaven å sette seg inn i besvarelsen.

- Dokumentasjon

Det skal legges ved dokumentasjon som beskriver utformingen av programmet (f.eks. klasse-diagram). Denne dokumentasjonen bør være så god at en som ikke er kjent med programmet eller oppgaven kan forstå rollen til hver klasse og hvordan hver metode fungerer uten å være nødt til å lese selve kildekoden.

- Pseudokode

Som en del av dokumentasjonen kan det være nyttig å legge ved pseudokode som gir en skisse over hvordan programmet fungerer.

- Kildekode

All kildekoden skal selvsagt legges ved. All kildekode **skal** inneholde javadoc-kommentarer til klasser og metoder.

- Innleveringsfrist

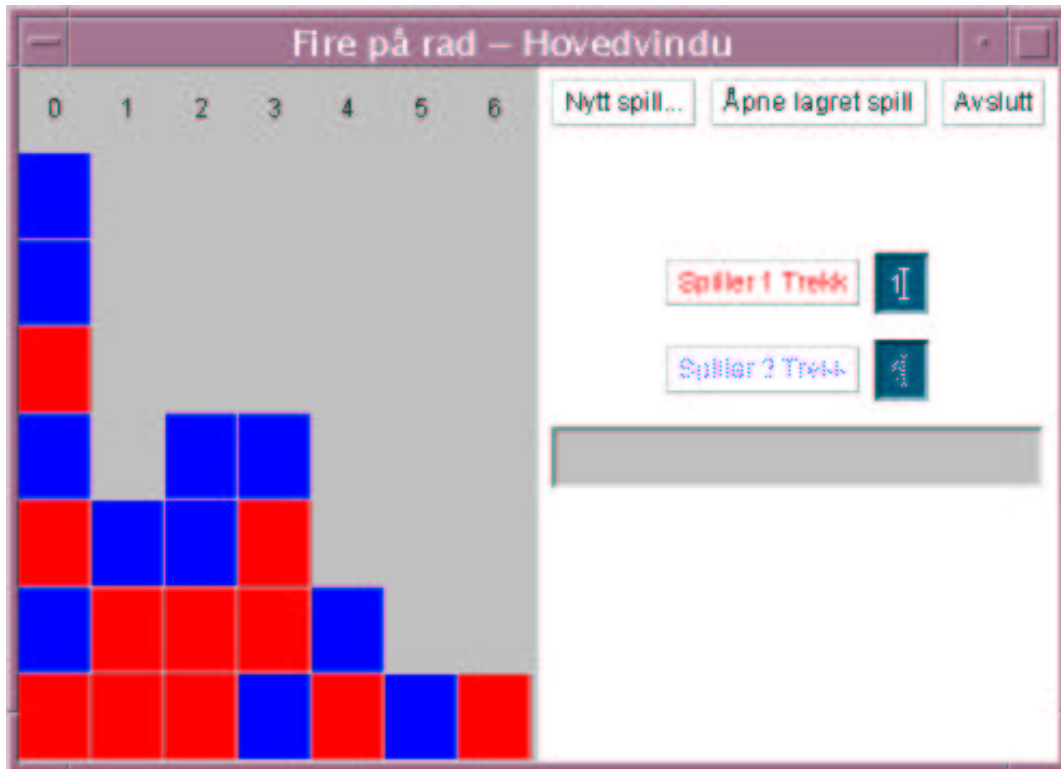
Innleveringsfristen er onsdag 28. november 2001, Kl 16:00. Innlevering etter fristen vil resultere i "ikke godkjent" oblig. Dersom man leverer innen fristen men ikke får godkjent, vil man få en anledning til å levere en forbedret utgave. Detaljer om hvordan innleveringen skal skje vil bli beskrevet på I110-websiden.

Et utvidet Fire-På-Rad (FPR) spill

Du skal utvide programmet som ble laget i den andre obligatoriske oppgaven. Målet er fremdeles å laget et program som skal la brukerne spille FPR. Nytt for denne oppgaven er:

- Programmet skal ha et grafisk grensesnitt fremfor et tekstbasert grensesnitt.
- Programmet skal kunne lagre tilstanden sin til fil og skal kunne lese en tidligere lagret tilstand fra fil.

Bortsett fra dette, skal det nye programmet tilby samme funksjonalitet som det gamle.



Grafisk grensesnitt

I denne oppgaven skal det tekstbaserte brukergrensesnittet fra obligatorisk oppgave 2 erstattes av et grafisk brukergrensesnitt. All interaksjon med brukeren av programmet skal nå foregå gjennom det grafiske brukergrensesnittet. Programmet skal ha et hovedvindu hvor det er mulig å velge følgende operasjoner:

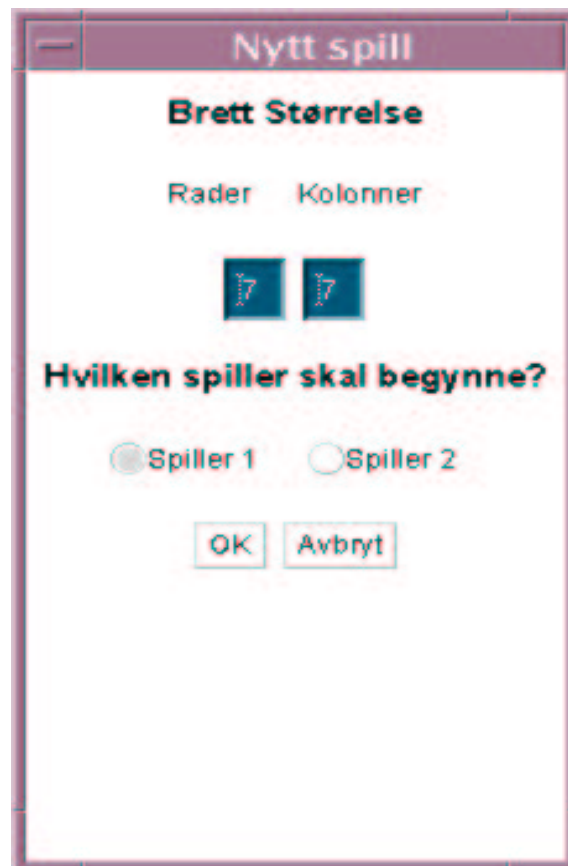
1. Starte en ny omgang av FPR. Ved oppstart av et nytt spill skal brukeren få opp et eget dialogvindu der det går an å velge:
 - Antall rader på spillebrettet
 - Antall kolonner på spillebrettet
 - Hvilken spiller som skal begynne spillet
2. Avslutte spillet
3. Anledning for spiller 1 og 2 til å gjøre et trekk
4. Åpne et lagret spill

For implementasjon av hovedvinduet vil det være naturlig å bruke klassen `JFrame`. Hovedvinduet vil være eier av de andre vinduene i programmet (dialogvinduene). Implementasjon av dialogvindu for nytt spill bør gjøres ved å bruke klassen `JDialog`.

Hovedvinduet skal vise en grafisk presentasjon av spillebrettet, slik at spillerene kan se hvordan brettets tilstand ser ut. Presentasjonen av spillebrettet kan implementeres ved hjelp av `JPanel` klassen. Ved å bruke layout manager klassen `GridLayout` kan en lage et rutenett som kan brukes til å "tegne" spillebrettet på. I hver rute kan en plassere et objekt av klassen `JLabel`. Objekter av klassen `JLabel` kan brukes til å plassere en tekst i en rute, eller en kan endre bakgrunnsfargen til

ruten med metoden `setBackground(Color bg)`. Denne metoden har klassen `JLabel` arvet fra klassen `JComponent`. For at bakgrunnsfargen til et `JLabel`-objekt skal vises, må en kalle `JLabel`-objektets `setOpaque` metode med verdien `true` som parameter.

Hovedvinduet må også ha et tekstfelt der det kan skrive ut feilmeldinger (f.eks. dersom en spiller prøver å gjøre et ugyldig trekk). Et eksempel på hvordan hovedvinduet kan utformes er gitt ovenfor.



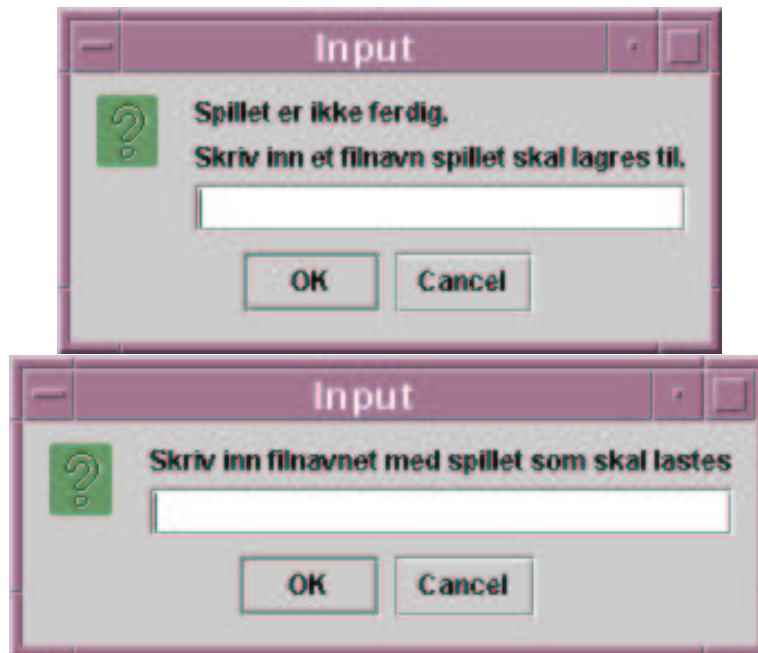
Lagring og innlasting av tilstand

Dersom man midt under en spilleomgang velger å avslutte spillet, skal programmet lagre spillets tilstand til en fil. Brukeren skal gis beskjed om dette ved at en får opp en egen “dialogboks” som gir beskjed om dette, og der en kan skrive inn et filnavn spillet skal lagres til.

Når en senere starter FPR, skal en kunne velge å laste inn det lagrede spillet ved å trykke på en knapp i hovedvinduet. Brukeren skal da få opp en dialogboks der en kan skrive inn filnavnet med spillet en ønsker å fortsette på. Slike dialogbokser kan lages ganske enkelt ved bruk av klassen `JOptionPane`.

Lagring og innlesing av spillets tilstand skal implementeres ved hjelp av objekt-serialisering. Den enkleste strategien for å implementere dette er å ha et objekt som inneholder all den relevante informasjonen i programmet (enten som instansevariabler eller referanser), slik at det eneste man trenger å gjøre er å serialisere/deserialisere dette objektet. Objekter av klasser som implementerer kontrakten (interfacen) `Serializable` kan enkelt skrives til/leses fra fil ved hjelp av klassene `ObjectOutputStream` og `ObjectInputStream`. Eksempel på bruk av objektserialisering er bl.a. gitt i

ukeoppgave 9.



Alternativ til bruk av egen implementasjon av oblig 2

Det vil bli lagt ut et løsningsforslag til obligatorisk oppgave 2 når rettingen av de innleverte oppgavene er ferdig. De som ønsker det kan bruke dette løsningsforslaget (til å bygge videre på) for å gjøre obligatorisk oppgave 3. Dette er et alternativ til å bruke sin egen implementasjon av oblig 2. Løsningsforslaget vil bli gjort tilgjengelig via I110-oppgavesiden senest 16. november. Dette løsningsforslaget vil også bli gjennomgått på fellesgjennomgangen den 16. november.