Tråder - Del I

IN1010 - Uke 9

Tobias Paulsen

Vår 2021

Tråder og parallell programmering

- Sekvens av instruksjoner i et program (”thread of execution”)
- Tidligere har vi skrevet sekvensielle program, de har en tråd (main) med en sekvens av instruksjoner
- Hvis vi ønsker at to eller flere oppgaver skal kjøre samtidig kan vi lage flere tråder. Disse kan da kjøres i parallell
- Ulike tråder kan dele på objekter (minne), og ”snakke” med hverandre. I IN1010 så skal ikke trådene kommunisere direkte, men de bruker delte objekter

Oppretting av tråder

- Kan se på en tråd som en arbeider, altså noe som kan utføre en oppgave. Disse arbeiderne er objekter av klassen `Thread`
- Oppgaven som trådene utfører er en sekvens av intruksjoner som er definert i klasser som inmplementerer grensesnittet `Runnable`
- Man må derfor ha **både** et objekt av `Thread` og et objekt av `Runnable`
- Man starter en tråd ved å kalle på metoden `start()`, da vil også Runnable sin medode `run()` bli kalt automatisk

```
interface Runnable {
  void run();
}
```

Eksempel

Skal se på et program som starter opp tråder som kan telle telle fra 1 til n

Synkronisering

- Hvis to eller flere tråder har tilgang på delt data kan kan støte på problemer
- Eksempel: ”shared counter”

Synkronisering

- Løsningen på dette er å lage en kritisk region (mutex), hvor kun en tråd har tilgang om gangen. andre tråder må da vente på sin tur
- Man lager en mutex med låser. Trådene låser av når de går inn og låser opp når de går ut av den kritiske regionen
- `java.util.concurrent.locks.Lock` er et grensesnitt med metodene `lock()` og `unlock()`
- Selve låseopgjektet lages av `java.util.concurrent.locks.ReentrantLock`
- `try, catch, finally`

Monitor

- God strategi for å beskytte dataen
- Monitor er et objekt som innkapsler den delte dataen, og definerer synkroniserte metoder for å jobbe med denne dataen
- I IN1010 brukes **alltid** monitorer, da dette er den mest objektorienterte måten å gjøre det på
- Fortsettelse på ”shared counter”

Conditions

- Tråder må vente på at en betingelse er oppfylt før den kan gjøre noe
- Ventekø tilknyttet en monitor. Venter på et signal slik at tråden kan fortsette
- `java.util.concurrent.locks.Condition`

```
interface Condition {
  void await();     // Venter på signal (eller interruption)
  void signal();    // Vekker en ventende tråd
  void signalAll(); // Vekker alle ventende tråder
}
```