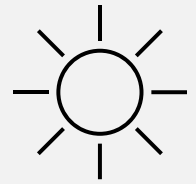
GRUPPE 11 IN1010 – UKE 10

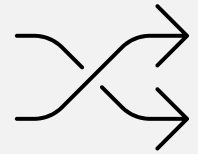
Sivert Fjeldstad Madsen

I DAG

- Repetisjon av tråder
- Ulike måter å vente på tråder
- Conditions igjen
- Livekode
- Oppgaver

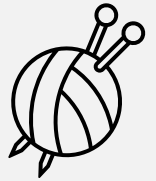


TRÅDER – KORT REPETISJON



- Vi bruker tråder for å utføre flere forskjellige ting *samtidig*
- Dette gjøres gjennom Java sin innebygde **Thread**-klasse
- Koden vi vil skal utføres skriver vi i **run()**-metoden til en klasse som implementerer interfacet **Runnable**
- Dersom flere tråder skal aksessere samme minneområde må vi lage en **monitor**
  - Inne i monitoren bruker vi **låser** for å begrense tilgangen til minneområdet
- Om vi vil at hovedprogrammet vårt skal vente på trådene, må vi (f.eks.) bruke **.join()**

VENTE PÅ TRÅDER



- Den vanligste (og enkleste) måten å vente på tråder på er å bruke **.join()**
  - Dette gjør at vi står og venter helt til en gitt tråd er ferdig
- Men kanskje vi vil samle alle trådene mer enn én gang?
- Eller kanskje vi vil at trådene skal vente på main?
- Da kan vi bruke **barrierer**
- I IN1010 er det to som er mest relevante:
  - **CountDownLatch** og **CyclicBarrier**

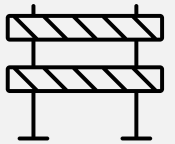
COUNTDOWNLATCH

- En **CountDownLatch** er litt som en «veisperring» som krever at et gitt antall tråder står og venter før den åpner seg

- Når man oppretter en **CountDownLatch** må man gi den et tall som representerer hvor mange ganger man må «telle ned» før bommen åpnes
- Dette kan brukes f.eks. for å gjøre at alle trådene må vente på et «startsignal» fra main-tråden før de setter i gang
- Man kan også bruke det om man vil at main-tråden skal vente til trådene er ferdige med *en del* av oppgavene sine

CYCLICBARRIER

- Fungerer ganske likt som en CountDownLatch, men kan gjenbrukes!
- Vi må fortsatt oppgi hvor mange tråder som må vente ved barrieren før den åpnes
- Egner seg godt dersom trådene skal utføre mange operasjoner som krever en eller annen form for synkronisering mellom hver operasjon
- Eller vi kan bruke det til å utføre noe etter at x antall tråder har gjort en oppgave
- Kan være litt knotete å holde styr på



CONDITIONS

- Ofte kan det være bedre å bruke Conditions enn barrierer
  - Husk at en condition opprettes via en lås!
- F.eks. for å gi et startsignal kan trådene vente ved en condition til main-tråden signaliserer at den er oppfylt
- Hvilken type venting eller signalisering du bør bruke kommer helt an på hva du faktisk prøver å gjøre
- Den aller beste formen for synkronisering er som oftest *ingen* synkronisering!
  - …men det er ikke alltid mulig å få til.

OPPSUMMERING

- Ikke så mye nytt denne uken
- Den viktigste barrieren vi trenger å tenke på er **.join()**
- Synkronisering er viktig, men det beste er å unngå det helt om mulig

- Ukas tips:
  - Debugger i VSCode