


Added by [krlang](#), last edited by [krlang](#) on Oct 21, 2013


 Unknown macro: 'metadata-list'

Basics

For å forstå hvordan tråder i java fungerer er det viktig å kunne skille mellom en **tråd** (*thread*) og en **prosess** (process). En prosess kan sees på som kjøring av et program eller en applikasjon. I operativsystemer (av typen multitasking) vil det ofte finnes flere prosesser, og hver enkelt har sitt eget minneområde.

En hver prosess har minst én tråd. En tråd er en "lettvektet"-prosess som kan kjøre kode i et program eller en applikasjon, men vil benytte seg av ressursene til prosessen den tilhører

Figuren under prøver å framstille (veldig enkelt) hvordan en prosess har en **hovedtråd** som utfører kode, og samtidig kan lage nye tråder som utfører kode parallelt. Alle tre trådene vil dele på ressursene (f.eks. minnet) som prosessen har fått tildelt.

 Unknown macro: 'lucidchart'

Man kan kanskje spørre seg; hvorfor ønsker man å utføre kode i forskjellige tråder?

Et vanlig bruksområde for flertrådsprogrammering er i applikasjoner hvor man har klient-tjener arkitektur. Det kan være lettere å forstå med en analogi;

Taxi Trondheim AS fungerer på følgende måte:

En døgnåpen taxisentral tar i mot henvendelser fra kunder, avklarer hvilken tjeneste kunden vil ha (vanlig taxi, maxitaxi, tidspunkt for henting, osv.), kontakter en passende taxi og overlater kunden til taxisjåføren. Taxisjåføren henter så kunden, og kjører dit kunden ønsker å dra. Når kunden er framme, slippes vedkommende av og taxisjåføren stopper for å vente på en ny oppringning fra taxisentralen. Taxi Trondheim AS har mange taxier som kan kjøre mange kunder samtidig.

I dette eksempelet vil hele Taxi Trondheim AS være en **prosess**. Taxisentralen som tar i mot henvendelser fra kunder tilsvarende **hovedtråden**, som kjører så lenge prosessen gjør. Når sentralen kontakter taxier (som bare står stille) og tildeler de kunder, tilsvarende dette at **hovedtråden** lager en ny **tråd**. Etter at taxisjåføren har utført oppdraget, kjørt kunden dit den ønsker, stopper taxisjåføren og tråden avsluttes. Men hovedtråden (taxisentralen) vil naturligvis fortsatt ta i mot henvendelser. En prosess (Taxi Trondheim) vil altså kunne mange tråder (taxier) kjørende samtidig.

Trådobjekter

For at et trådobjekt skal kunne brukes trenger det egentlig bare to ting:

Først må vi fortelle tråden hva den skal utføre i **run**-metode. Deretter må tråden **startes**.

Det er hovedsakelig to måter å starte en ny tråd på.

Gi et **Runnable**-objekt til en ny **Thread**:

Klassen Taxi implementerer **runnable**, et grensesnitt som krever at taxi har **run**-metoden. Main-metoden lager så nye tråder, og gir de et **runnable**-objekt som forteller hvordan tråden skal kjøres. Deretter kan trådene startes. Denne måten er mer generell fordi runnable-objektet kan arve fra en annen klasse enn Thread.



```

Runnable {

    int seats;

    public Taxi(int seats) {
        this.seats = seats;
    }

    public void run() {
        // Hente kunde
        // Kjøre kunde
        // Slippe av kunde
    }
}

```

```

args) {

    Taxi taxi = new Taxi(4);
    Thread trd1 = new
    Thread(taxi);
    trd1.start();

    Taxi maxitaxi = new Taxi(16);
    Thread trd2 = new
    Thread(maxitaxi);
    trd2.start();
}

```

Instansere en klasse som arver **Thread**:

Klassen taxi arver **Thread** og kan derav startes selv. Man legger oppførselen (**run**-metoden) til tråden direkte. Denne måten er lettere å bruke i enkle applikasjoner, men begrenser klassen ved at den *må* arve **Thread**.

Taxi-Thread

```

public class Taxi extends
Thread {

    int seats;

    public Taxi(int seats) {
        this.seats = seats;
    }

    public void run() {
        // Hente kunde
        // Kjøre kunde
        // Slippe av kunde
    }
}

```

Main-Thread

```

public static void main(String[]
args) {

    Taxi taxi = new Taxi(4);
    taxi.start();

    Taxi maxitaxi = new Taxi(16);
    maxitaxi.start();
}

```

Med tråder følger det mye nyttig funksjonalitet, f.eks. `Thread.sleep()` -funksjonen. Mer om funksjonalitet i forbindelse med tråder finnes [her](#).

Like Be the first to like this

Labels sidetype-teori

1 Child Page

Tråder, eksempel med flere tråder