Trådar

Objekt-orienterad programmering och design

Alex Gerdes, 2018

Trådar

- Många program måste kunna hålla på med flera saker samtidigt:
  - bokningssystem av olika slag
  - en webbserver som måste kunna leverera flera webbsidor samtidigt
  - en grafisk applikation som ritar samtidigt som den arbetar med nästa bild
  - ett grafiskt användargränssnitt som utför beräkningar och samtidigt är redo att ha dialog med användaren
- Denna typ av program kallas samverkande program (concurrent programs).
- I Java beskriver man aktiva objekt (och parallellism) med hjälp av standardklassen `Thread`. Aktiviteter som pågår samtidigt kallas därför i Java för trådar.

Icke-deterministiskt beteende

- Sekventiella program säges vara deterministiska vilket betyder att:
  - vilka operationer som utförs i programmet är en konsekvens av föregående operationer
  - det är förutsägbart i vilken ordning operationerna i ett sekventiellt program kommer att utföras
  - Om ett program körs flera gånger med samma indata kommer samma resultat att erhållas varje gång
- Program med parallellism uppvisar ett icke-deterministiskt beteende:
  - I vilken ordning operationerna sker mellan de olika programflödena vet vi inte
  - två programkörningar med samma indata kan ge olika resultat

Övning

- Börja från koden som finns att ladda ner från hemsidan.
  - Vår gamla kod från förra veckan är uppdaterad enligt vad vi gjorde på förre föreläsningen. Gå igenom koden och inspektera implementationen av den Decorator och Chain of Responisbility designmönster.

- Vi har lagt till en extra klass `Creator` som kör i en tråd och lägger till polygoner.
  - I DrawPolygons finns kod som skapar och startar en Creator tråd men tyvärr funkar det inte än. Fixa detta. Tips: kom ihåg varför den andra trafikljus i Signal projektet började inte blinka.
- Skapa en klass `Terminator` som kör in sin egen tråd och tar bort en polygon. Meningen är att vi kommer att lägga till en polygon och sedan ta bort en, dvs i **samma takt**.
  - I modellen finns nu också en metod (`removePolygon`) för att ta bort en polygon.
  - Kör programmet och kolla om det kör utan problem, dvs efter varje addition kommer en reduktion.
  - Troligen kör programmet inte utan problem och addition/reduktion sker inte i samma takt. Dessutom kan det hända något `very bad'. Det kan vara så att vi försöker att ta bort ett element från en tom lista.
  - Lägg till en koll i `removePolygon` metoden (som första sats i metoden) som väntar tills listan är inte tomt och sedan fortsätter: `while (polygons.isEmpty()) {}` (while-bodyn ska vara empty alltså) Lägg till en koll i `addPolygon` metoden som väntar tills listan är tomt och sedan fortsätter: `while (!polygons.isEmpty()) {}`
  - Kör programmet igen, det kan hända vi hamnar i en 'Deadlock' situation, dvs creatorn och terminatorn väntar på varandra.
  - Försök att åtgärda deadlocken med hjälp av metoderna `wait()` och `notifyAll()`, kolla API på nätet.
- För mer utmaning: istället för begränsa att ha max *en* polygon i listan kan man definiera en viss kapacitet och se till att vi inte skapa fler än så. Man ska släppa kravet att det ska gå i takt och sedan skapa flera trådar som skapa och radera polygoner.