

Plenum: Rekursjon

Ahmed Waseem Saeed

Mars 2024

Introduksjon

Velkommen til ny plenumstime! Denne uken skal vi lære om et av de mest sentrale konseptene innenfor informatikk, nemlig rekursjon. Vi skal se hvordan rekursjon lar oss løse problemer på en elegant måte. Som oppvarming, så skal vi implementere hent-metodene fra de to forrige ukene om beholdere (AListe) på nytt, ved hjelp av rekursjon. Deretter skal vi modellere et filsystem ved hjelp av filer og mapper, og se hvordan vi rekursivt kan skrive ut innholdet.

Du står fritt frem til å implementere hjelpemetoder der du trenger det. Med rekursive metoder er det ofte en fordel å ha hjelpemetoder.

Oppvarming: AListe

De to foregående ukene jobbet vi med AListe og ulike metoder som opererer på lista. Vi kan faktisk skrive mange av metodene rekursivt, og i dag skal vi skrive hent() og hentPaaIndeks() ved hjelp av rekursjon.

Utfordring: leggTil() kan også skrives rekursivt, prøv selv på dette hvis du har tid.

Filsystem

Vi skal nå modellere et filsystem ved hjelp av filer og mapper. Mapper er en form for filer, og det er naturlig at vi benytter arv her for å modellere dette. Rekursjon er en naturlig måte å traversere et filsystem på, og det er nettopp dette vi skal modellere. Vi skal skrive rekursive metoder som lar oss finne en fil fra en gitt mappe, finner hele stien for en gitt fil eller mappe, og skriver ut alt innholdet i mappen under seg. Først og fremst må vi skrive Fil- og Mappe-klassen.

Fil

Klassen Fil har følgende instansvariabler: navn (som indikerer navnet til fila eller mappen) og en boolsk variabel erMappe som settes til False. I tillegg har alle filer en foreldremappe av typen Mappe. Denne skal også tas inn i konstruktøren. Den aller første mappen i et filsystem vil ha null som foreldremappe. *Hvorfor er dette viktig med tanke på rekursjon?*

Skriv også hent-metoder for instansvariablene. Deretter skal du overskrive toString-metoden til å skrive ut navnet på fila, og om fila er en mappe eller ikke.

Mappe

Klassen Mappe arver fra Fil. Klassen skal i konstruktøren sette erMappe til True. I tillegg har klassen en liste over filer (inkludert mapper) som eksisterer i den gitte mappen. Nå har vi strukturen til klassene på plass, og kan begynne å skrive metoder som lar oss hente ut informasjonen vi trenger i et filsystem, ved hjelp av rekursjon.

Tips til de neste oppgavene: Tenk nøye gjennom hva som er base-tilfellet, samt hvordan vi skal sette i gang den rekursive metoden.

Skriv den rekursive metoden finnFil(String filnavn) som tar inn et filnavn. Denne metoden skal lete gjennom alle filer og mapper i mappen og returnere det første Fil-objektet med likt navn. Metoden skal lete gjennom sine undermapper (eng: subdirectories) rekursivt til den finner riktig fil.

Skriv metoden finnAbsoluttSti() i klassen Fil, som returnerer en streng som angir alle mapper som leder til denne fila (skilt med "/"). Denne skal bruke foreldremappen til å finne ut den absolutte stien som leder til fila (eller mappen).

Til slutt skal du skrive metoden skrivUtInnhold() i Mappe. Denne metoden skal skrive ut alt innhold i mappen ved å printe ut navnet på mapper og filer i sin egen mappe, før den kaller rekursivt på sine undermapper. Merk at du vil skrive ut innholdet i den relative stien også, altså inkludere navnet på mappen man kom fra. For å få til dette kan det være nyttig med en hjelpemetode som setter i gang rekursjonen med en tom streng.

Et kall på denne metoden (i mappe1) skal eksempelvis skrive ut på en slik måte:

- mappe1/fil1.txt
- mappe1/mappe2/fil4.txt
- mappe1/mappe3

Etter å ha implementert denne metoden har du faktisk modellert noe som ligner kommandoen "ls -R", som printer ut alt innholdet i en gitt mappe og dens undermapper. Bra!

For de spesielt interesserte

Call-stacken ved flere rekursive kall kan bli ganske stor. For å håndtere dette og unngå å legge til en ny stack frame på call-stacken, så kan man benytte noe som heter halerekursjon. Ta en titt på denne [wiki-artikkelen](#) dersom du er interessert i å lære mer. Et vanlig mønster for å gjøre en "vanlig" rekursiv metode om til en metode som benytter halerekursjon, er å ha en hjelpemetode som setter i gang rekursjonen. Her bruker man ofte en akkumulator-verdi som endres inne i funksjonskallene fremfor å vente på returverdien før man utfører en beregning.

Som en øvelse så kan du prøve å finne summen av en AListe ved hjelp av hale-rekursjon, der verdiene er heltall.