F Enumtyper

F.1 Et enkelt eksempel

Dette er en kort innføring i enumtyper. Ønsker du å få en dypere forståelse, bør du gå til «kilden», dvs. til §8.9 i JLS8 (The Java® Language Specification, Java SE 8 Edition).

Det er mange situasjoner der en «datatype» bare kan ha bestemte verdier. F.eks. kan vi se på *Ukedag* som en datatype med syv faste verdier, dvs. mandag, tirsdag, osv. En ukedag kan representeres ved en tegnstreng ("mandag", "tirsdag", osv.) eller med et tall (f.eks. en *int*) fra 1 til 7. Men det er bedre (og sikrere) å definere en enumtype *Ukedag*:

```
public enum Ukedag
{
  Mandag, Tirsdag, Onsdag, Torsdag, Fredag, Lørdag, Søndag
}
```

Programkode **F.1** *a)*

I enumtypen er det satt opp syv *enumkonstanter*. De oppfører seg som om de skulle oppfylle *public static final*. De er med andre ord offentlige og aksesseres ved hjelp av typenavnet:

```
Ukedag dag = Ukedag.Lørdag;  // en referanse
System.out.println(dag);     // Utskrift: Lørdag
```

Programkode **F.1** *b)*

En enumtype er en *referansetype* på lik linje med klasser, grensesnitt og tabeller. Det betyr at den er en subtype til `Object`. En enumkonstant (som er en instans av enumtypen) har da bl.a. metodene *toString*(), *equals*(), *hashCode*() og *getClass*(). Det er *toString*() som implisitt kalles i utskriftssetningen i over. Den er overskrevet (eng: overridden) i Ukedag.

En enum har noen få statiske metoder. Den mest brukte heter *values*() og returnerer en enumtabell. Den inneholder alle konstantene (i samme rekkefølge som de har i enumtypen):

```
for (Ukedag dag : Ukedag.values())
{
  System.out.print(dag + " ");
}
// Utskrift: Mandag Tirsdag Onsdag Torsdag Fredag Lørdag Søndag Lørdag
```

Programkode **F.1** *c)*

Hvis en har navnet på en enumkonstant som en tegnstreng (*String*), så kan en finne den tilhørende konstanten ved hjelp av den statiske metoden *valueOf*():

```
Ukedag dag = Ukedag.valueOf("Søndag");
System.out.println(dag);  // Utskrift: Søndag
```

Programkode **F.1** *d)*

En enumtype er *Comparable*, dvs. `E` *implements Comparable*<E> for enhver enumtype `E`. Det er den rekkefølgen konstantene er satt opp i som bestemmer ordningen. Kall mandag til fredag for arbeidsdager. I flg. Ukedag-tabell er de tilfeldig satt opp. Men de kan sorteres:

```
Ukedag[] arbeidsdager =  // en Ukedag-tabell som inneholder "arbeidsdager"
{
  Ukedag.Torsdag, Ukedag.Onsdag, Ukedag.Mandag, Ukedag.Fredag, Ukedag.Tirsdag
};

Arrays.sort(arbeidsdager);  // bruker at Ukedag er Comparable
System.out.println(Arrays.toString(arbeidsdager));

// Utskrift: [Mandag, Tirsdag, Onsdag, Torsdag, Fredag]
```

Programkode* F.1 *e)

En enumkonstant bør oppfylle den vanlige normen for konstanter, dvs. kun store bokstaver. I enumtypen Ukedag er det kun stor forbokstav. Noen ganger dropper man endestavelsen *dag* og bruker isteden Man, Tirs, Ons, osv. I slike tilfeller passer det kanskje best med kun store bokstaver - MAN, TIRS, ONS. Vi gjør om enumtypen Ukedag slik at konstantene får disse korte navnene. Men vi ønsker fortsatt at en utskrift skal gi hele navnet på dagen. Det kan vi få til ved å gi enumtypen Ukedag en privat instansvariabel *navn*. Da må Ukedag også ha en konstruktør for at *navn* skal kunne få verdi:

```
public enum Ukedag
{
  MAN ("Mandag"),      // konstruktøren brukes
  TIRS ("Tirsdag"),    //  -   "   -   "   -
  ONS ("Onsdag"),      //  -   "   -   "   -
  TORS ("Torsdag"),    //  -   "   -   "   -
  FRE ("Fredag"),      //  -   "   -   "   -
  LØR ("Lørdag"),      //  -   "   -   "   -
  SØN ("Søndag");      //  -   "   -   "   -

  private final String navn;  // instansvariabel
  private Ukedag(String navn) { this.navn = navn; }    // konstruktør

  @Override public String toString() { return navn; }  // ny versjon
}
```

Programkode* F.1 *f)

Den nye versjonen (override) av *toString*() gjør at en utskrift vil bli som før:

```
System.out.println(Arrays.toString(Ukedag.values()));
// Utskrift: [Mandag, Tirsdag, Onsdag, Torsdag, Fredag, Lørdag, Søndag]
```

Programkode* F.1 *g)

Men vi har heldigvis også muligheten til å få en utskrift med de korte navnene. En enumtype har instansmetoden *name*() og den gir alltid det «synlige» navnet på konstanten:

```
StringJoiner s = new StringJoiner(", ", "[", "]");
for (Ukedag dag : Ukedag.values()) s.add(dag.name());
System.out.println(s.toString());  // [MAN, TIRS, ONS, TORS, FRE, LØR, SØN]
```

Programkode* F.1 *h)

Obs: Det er ikke mulig å lage enumkonstanter på noen annen måte enn slik det gjøres i *Programkode F.1 f)*. Det skjer ved hjelp av konstruktøren. Det ser enklere ut i *Programkode F.1 a)*, men også der brukes en konstruktør, dvs. den implisitte standardkonstruktøren. Men den virker ikke lenger i *Programkode F.1 f)*. Men det er mulig å legge inn en parameterløs konstruktør. Men den måtte i så fall sette *navn* til *null* eller til en fast verdi. Se *Oppgave x*. Operatoren **new** kan ikke brukes her.

I *Programkode F.1 e)* laget vi en tabell for «arbeidsdager». En slik tabell kan være nyttig flere steder. Vi kan derfor la enumtypen Ukedag ha en statisk metode som returnerer tabellen. I tillegg kan vi lage metoder for *hverdager* (mandag - lørdag), *fridager* (lørdag og søndag) og *helligdag* (søndag):

```
public static Ukedag[] arbeidsdager()  // mandag til fredag
{
  return new Ukedag[] { MAN, TIRS, ONS, TORS, FRE };
}

public static Ukedag[] fridager()      // lørdag og søndag
{
  return new Ukedag[] { LØR, SØN };
}

public static Ukedag[] hverdager()     // mandag til lørdag
{
  return new Ukedag[] { MAN, TIRS, ONS, TORS, FRE, LØR };
}

public static Ukedag helligdag()       // søndag
{
  return SØN;
}
```

Programkode F.1 i)

Mandag regnes som ukedag 1, tirsdag som ukedag 2, osv. Men er det mulig å finne ukedagen til en en vilkårlig ukedagkonstant? Enumtypen Ukedag har instansmetoden *ordinal*(). Den returnerer konstantens nummer i oppramsingsrekkefølgen. Den første er imidlerntiid nr. 0, den neste nr. 1, osv. Dette passer ikke helt vårt behov. Men vi kan bruke den til å lage en ny instansmetode *dagnr*() i Ukedag:

```
public int dagnr() { return ordinal() + 1; }  // hører til enum Ukedag
```

Hvis du har lagt denne metoden (og metodene i *Programkode F.1 i*) inn i enum Ukedag, vil flg. programsetning virke:

```
System.out.println(Ukedag.helligdag().dagnr());  // Utskrift: 7
```

Det kan oppstå et problem med metoden *dagnr*(). Det kunne tenkes at det senere var ønskelig å sette opp enumkonstantene i en annen rekkefølge enn de nå har. Eller at det kunne være to konstanter for samme dag. F.eks. kan det ikke hete lørdag på nynorsk. Der heter det laurdag. Vi kunne da legge inn en konstant LAUR (med navn Laurdag) mellom lørdag og søndag. Men da vil programkoden over gi 8 som svar istedenfor 7.

En bedre (sikrere og mer fremtidsrettet) løsning er å gi enumtypen Ukedag en privat instansvariabel av typen *int* med navn *dagnr*. Da må konstruktøren utvides med en *int*-parameter. Starten på Ukedag blir da slik:

```
public enum Ukedag
{
  MAN ("Mandag", 1),      // den nye konstruktøren
  TIRS ("Tirsdag", 2),    //  -   "   -   "   -
  ONS ("Onsdag", 3),      //  -   "   -   "   -
  TORS ("Torsdag", 4),    //  -   "   -   "   -
  FRE ("Fredag", 5),      //  -   "   -   "   -
  LØR ("Lørdag", 6),      //  -   "   -   "   -
  SØN ("Søndag", 7);

  private final String navn;  // instansvariabel
  private final int dagnr;    // instansvariabel

  private Ukedag(String navn, int dagnr)  // den nye konstruktøren
  {
    this.navn = navn; this.dagnr = dagnr;
  }

  public int dagnr() { return dagnr; }
  @Override public String toString() { return navn; }

  // resten av metodene
  // . . . . . .
  // . . . . . .

} // Ukedag
```

Programkode* F.1 *j)

Oppgaver til Vedlegg F.1

1. Ta utgangspunkt i Programkode F.1 j) og gjør enumtypen Ukedag helt ferdig. Lag så de boolske instansmetodene *erHelligdag*(), *erArbeidsdag*(), *erFridag*() og *erHverdag*().
2. Lag enum Måned der du bruker de tre første bokstavene som navn på enumkonstantene, dvs. JAN, FEB, MAR, osv. De fulle navnene skal vær Januar, Februar, osv. Månedene skal ha et månedsnr fra 1 til 12. Definer også hva som er vår, sommer, høst og vinter. Lag de boolske instansmetodene *erVårmåned*(), *erSommermåned*(), osv.