**Observatør-observert:**

Nyttig hvis det er viktig at objekter er konsekvente med hverandre. Et eksempel kan være highscore i et spill. Her må objektet som inneholder listen av «scores» si ifra til objektet som kontrollerer brukergrensesnittet slik at bruker kan se en oppdatert list.

For å være mer konkret må den observerte:

* Kunne endre tilstand
* Registrere observatøren
* Si i fra om endringer i tilstand

Og observatøren må:

* Kunne ta imot beskjeder om endringer av tilstand fra den observerte

Tilbake til highscore eksemplet, her må den observerte (lista) ha mulighet til å registrere observatøren, få nye «scores», og om den får noen nye «scores» skal den si i fra til observatøren om at en endring har skjedd. Mens observatøren skal ta imot beskjeder om endringer i lista, og oppdatere brukergrensesnittet.

**Delegering:**

Her har vi et objekt som har et sett metoder, og tar i bruk et annet objekt for å utføre logikken. En typisk bruk av dette er 1-n-assosiasjon. Et eksempel på hvordan jeg har brukt dette er at jeg hadde et *Population* objekt som inneholdte rundt 500~ *Body* objekter. For hver tidsenhet skulle hele populasjonens posisjon oppdateres, så jeg kalte en funksjon *update()* på *Population* objekter, som igjen kaller *update()* på hvert body objekt. *Population* objektet delegerer *update()* til *Body*