Singleton

Singleton-mønstret sikrer at kun en instans kan laves af en klasse, og giver global adgang (global synlighed) til denne. Dette er en fordel, hvis vi kun ønsker netop én instans af en klasse, hvilket vi har benyttet bl.a. ved alle vores container-klasser. De klasser hvor vi har benyttet singleton-mønstrer, er illustreret vha. metoden getInstance() på design klassediagrammet. I nogle tilfælde har dog singleton-mønstret ikke egnet sig fx ved klassen item. Item har nemlig en ArrayList over unit-objekter, hvilket klassificerer den som en slags container, men i dette system skal det være muligt at oprette mange item-objekter, og derfor vil det være logisk at undlade singleton-mønstret i denne her situation. Derimod egner fx singleton-mønstret til visse container-klasser fx EmployeeContainer til employee-objekter, idet vi ikke behøver mere end én til at samle vores medarbejdere.

Her er et eksempel på hvorledes Singlenton-mønstret er implementeret i programmet:

|  |
| --- |
| 14 public class SaleContainer {  15  16 private static SaleContainer instance; //en instans af selve klassen  17 private ArrayList<Sale> saleList;  18  19 /\*\*  20 \* Konstruktør  21 \* Initialiserer en tom database(saleList)  22 \*/  23 private SaleContainer() {  24 saleList = new ArrayList<Sale>();  25 }  26  27 /\*\*  28 \* En metode, der tjekker om der er oprettet en instans af klassen. Hvis dette er tilfældet, oprettet en ny.  29 \* **@return** En instans af klassen  30 \*/  31 public static SaleContainer getInstance() {  32 if (instance == null) {  33 instance = new SaleContainer();  34 }  35 return instance;  36 } |

Beskrivelse design klassediagrammet

Design klassediagrammet viser et overblik over det udviklede system. Det tager udgangspunkt i use-case-diagrammerne, domænemodellen og interaktions/system sekvensdiagrammerne. Disse nævnte elementer giver udviklerne en idé om hvilke dele der kunne indgå i sådan et systemet, heraf klasser, attributter og metoder. På samme måde som med domænemodellen, skaber design klassediagrammet et udgangspunkt, som udviklerne af systemet kan forholde sig til.