DD2385

Programutvecklingsteknik

Bilder till föreläsning 5

10/4 2014

Innehåll

- Introduktion till designmönster
- Designmönster:
  - MVC
  - Singleton

Designmönster

- Återanvändbar lösning på vanligt förekommande problem.
- 1995
- Boken *Design patterns*
- Gamma, Helm, Johnson, Vlissides → GoF

Designmönster

Några citat ur *Xiaoping Jia*:
Object-Oriented Software Development Using Java

- "Capture and **document** the **experience** acquired in software design in a relatively small number of design patterns to help designers aquire design expertise."
- "**Support reuse** in design and boost confidence in software systems that use established design patterns that have proven effective."
- "Provide a **common vocabulary** for software designers to communicate about software design."

Boken *Design Patterns*

Mönstren delas in i tre kategorier:

- Structural Creational Behavioral

Varje mönster beskrivs med

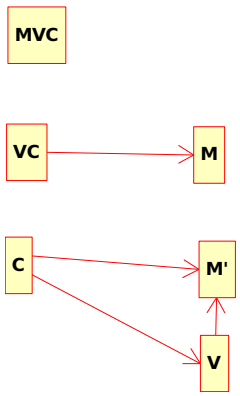
- Namn
- Kategori
- Syfte
- Andra namn på samma mönster
- Användningsområde
- Struktur (UML-klassdiagram)
- Deltagare (klasser och objekt)
- Exempel med kod i Java eller C++

Flera designmönster finns i Javas klassbibliotek

Model-View-Control

- Princip för strukturering av program(delar) med grafisk interaktion
- Model för data, logik, algoritmer (ingen grafik)
- View för grafisk representation av Model
- Control för användarens interaktion som uppdaterar Model (och View).
- Design-mönster eller arkitektur ?
- MVC är inte med i GoF

Model-View-Control



MVC

- Inte alltid rätt att skilja på View och Control
- T.ex. om View består av knappar och Control består av samma knappar
- M – VC, modellen för sig och View-Control tillsammans är vanligt
- Varje M, V, C kan bestå av flera klasser.

Vi tittar på några små programexempel under föreläsningen

Förebygg fel genom god design

Exempel:
Det får bara finnas en instans av klassen S.

- ett filsystem
- en fönsterhanterare
- ett kontrollobjekt

Hur gör man ?

Lösning 1:

Kom ihåg:
"Skapa bara ett objekt av S, gör `new S()` endast en gång!"

Lösning 2: *mycket bättre*

- Bygg in i designen av S att det bara får finnas en instans
- Program som försöker skapa flera går inte ens att kompilera

..... men hur gör man ?

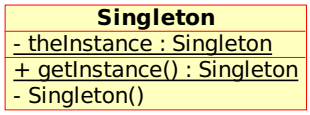
- I Java kan numera Enum-typer lösa problemet.
- I andra språk: mönstret Singleton
- Vi tar upp Singleton i Java ändå,
  som intressant användning av `private` och `static`
  och förberedelse till andra mönster.

Singleton - klass som bara har en instans

- Gör konstruktorn `private`
- Skapa objektet inuti klassen
- Användare får tag på det enda objektet genom metodanrop

Singleton

Det får bara finnas ett objekt av klassen



Går att utvidga till

"Det får finnas exakt N objekt av klassen"

Javakod för Singleton, alternativ 1

Det enda objektet skapas när klassen laddas av JVM.

```
public class Singleton {

    private static Singleton theInstance
        = new Singleton();

    private Singleton () {}

    public static Singleton getInstance(){
        return theInstance;
    }
}
```

Javakod för Singleton, alternativ 2

Här skapas det enda objektet när det efterfrågas första gången.

```
public class Singleton {

    private static Singleton
                    theInstance = null;

    private Singleton () {}

    public static Singleton getInstance(){
        if (theInstance == null)
            theInstance = new Singleton();
        return theInstance;
    }
}
```

Kan Singleton-egenskapen ärvas?

Nej, inte i Java!

- Konstruktorer ärvs inte (i Java)
- Privata variabler och metoder ärvs inte (i Java)

⇒ Singleton-egenskapen kan inte ärvas

Mock Object

Lite olika definitioner finns!

1. Ett interface implementeras med liten funktionalitet
2. Ett interface implementeras med simulerad funktionalitet