Objekt orienterad design: F8

Ch. 6.1 Dividing the code into packages

Första delen är alltid domän namnet baklänges.

se.kth

Sedan något som unikt identifierar produkten unikt inom organizabionen. typ avdelning och/eller projekt. ivl350.carRental. Sedan en del som identifierar ett specifikt package i projektet. model. Är lagret stort, kan man dela upp det i subpackages, payment.

se.kth.ivl350.carRental.model.payment

Vissa metoder som ej kan placeras i ett package, kan läggas i "utility class". Oftast är den ej app specifika.

6.2 - Code Convention.

Viktigt att koden är lätt att förstå. Flera personer kommer troligtvis att arbeta med koden.

[Table ...] Vanliga Java namn-konventioner.

6.3 - Comments

Det ska finnas javadoc kommentarer för varje deklaration som tillhör ett publikt gränssnitt.

```
/**
  Kommentar.
*/
```

Den ska beskriva vad den gör, men ej hur.

Hur tillhör implementationen, ej publikt gränssnitt.

Är det en metod ska den förklara

Parametrar @param
Return value @return

[Listing ...]
[Bild ...]

<Code> för Java keywords, namn, kod ex.

@link Om koden deklareras någon annanstans

Det är sällan meningsfullt att kommentera i metoder. Bättre byta ut till metoder som förklarar vad som sker.

6.4- Code Smell and Refactoring

Oönskade tillstånd på koden, tas bort genom att förändra koden utan att ändra dess funktionalitet

Duplicated code: Upprepning av kod, svår att underhålla, onödigt mycket kod.

Long Method: Metodens namn ska berätta allt den gör, behöver man kommentera i metoden, är den troligtvis för lång.

Large Class: Har den dålig sammanhållning, genom att flytta saker som har en nära relation till en egen klass får man bättre sammanhållning.

Long Parameter List: Svåra att förstå svårt att komma ihåg meningen av varje parameter.

Ändrar man parameter ändrar man publika gränssnitet. Använd datahållare.

99

Excessive use of primitive variables.

Onödigt många primitiva variabler gör klasser!

En array ska endast innehålla element av samma typ

Använd Enum istället för String.equals("SUCESS")

Meaningsless Names: Undvik meningslösa namn till varje kostnad. Använd inte en bokstav för identifierare

```
Person p = new Person();
```

Utan använd

```
Person person = new Person();
```

Förutom på tex x, y - koordinater och loopar.

Undvik tmp/temp

Använd inte numrering. account1 / account2

Var rädd för långa namn.

Anonymous Values: Alla värden ska ha förklarande namn. Bättre namn istället för att kommentera.

Bra att använda metoder för att namnge ett värde, tex på if-satser.