Labra 3: Refaktorointi

Labran tarkoituksena on harjoitella refaktorointia esimerkkisovelluksen kanssa.

Harjoitus tehdään käyttämällä pariohjelmointia!.

Esimerkkisovelluksesta on kaksi versiota, voit valita kummalla haluat tehdä.

- Javalla tehty toteutus, josta NetBeans 6.9 projekti.
- C#:lla tehty toteutus, josta Visual Studio 2010 projekti.

Esimerkkisovellus: Videovuokraamo

- Sovellus laskee vuokrattujen elokuvien hinnan ja tulostaa kuitin asiakkaalle.
- Elokuvan vuokraushintaan vaikuttaa elokuvan tyyppi ja kuinka pitkäksi aikaa elokuva vuokrataan.
- Elokuvatyyppejä on tällä hetkellä kolmenlaisia: normaaleja, uutuuksia ja lastenelokuvia.
- Sovellus laskee myös toistuvista vuokrauksista saatavia lainauspisteitä, joihin vaikuttaa elokuvan tyyppi.

Nyt sovellukseen täytyy lisätä seuraavat toiminnallisuudet:

1. Kuitti pitää pystyä tulostamaan xml-muotoon.
2. Uusi elokuvatyyppi: Klassikot
    - Klassikoiden hinta on 2 €
    - Jokaista klassikon vuokrauspäivää kohden saa yhden lainauspisteen.

Vaikkakin esimerkkisovellus on pieni ja refaktoroinnin hyöty saattaa tuntua turhalta, kuvittele että kyseessä on pieni osa suurempaa järjestelmää, jota tullaan myös myöhemmin muuttamaan.

Harjoituksen tarkoituksena ei ole uuden toiminnallisuuden toteuttaminen, vaan vanhan koodin refaktorointi siten, että uusi toiminnallisuus on helppo toteuttaa.

Refaktoroinnin helpottamiseksi on tehty yksinkertainen testi, joka testaa kuitin tulostusta ja vertaa tulostusta kovakoodattuun versioon.

Ohjeita:

- Tee pieniä muutoksia kerralla: muuta jotain, buildaa, testaa! jatka eteenpäin.
- Kirjaa ylös tehdyt refaktoroinnit ja uudet refaktoroinnit
- Pariohjelmointia! max 15min välein vaihdetaan näppäimistöä!

- Esimerkit harjoitusvaiheissa mainittuihin refaktorointeihin löytyy: http://www.refactoring.com/catalog/index.html
- Tutustu IDEn tarjoamiin refaktorointityökaluihin! Valmiina pitäisi löytyä ainakin seuraavat:
    - Rename
    - Extract interface
    - Encapsulate field
- Tutustu ohjelmakoodin, käännä se ja aja testit!
- Tutustu refaktotointiesimerkkeihin, etsi niistä seuraavat:
    - Extract Method
    - Move Method
    - Replace Temp with Query
    - Form Template Method
    - Replace Conditional with Polymorphism
    - Replace Type Code with State/Strategy
    - Self Encapsulate Field
- Refaktorointia tehdessä, huomio kaikki mahdolliset ongelmakohdat koodista.
    - Nimeämiskäytännöt, kommentit ja vanhat toteutustavat!

Harjoitusvaiheet:

Refaktoroinnin saa tehdä vapaasti, seuraavaksi on listattu esimerkkisovelluksen ongelmia ja refaktorointi ehdotuksia joilla pääsee alkuun.

1. Statement metodin osittelu ja purku

Statement metodissa näyttää olevan eniten ongelmia. Metodi rikkoo huolella SRP sääntöä.

Metodissa on sekaisin bisneslogiikkaa ja esitystason logiikkaa, jotka täytyy erottaa.

Jotta kuitin tulostaminen xml muotoon onnistuu, ilman duplikaattikoodia, on statement metodi purettava osiin.

1. Käytä ”Extract Method” refaktorointia metodin osiin jakamiseksi.
    - Vuokratun elokuvan hinnan laskeminen.
    - Lainauspisteiden laskeminen.
    - Laskun tulostus

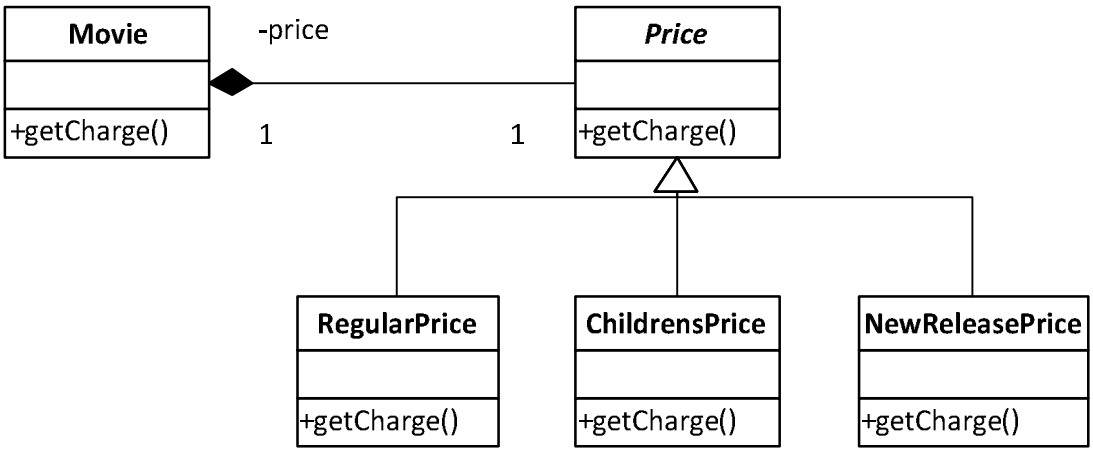
2. Vuokrauksen hinnan laskemisessa ei käytetä asiakkaan tietoja, silti logiikka sijaitsee Customer luokassa?
    - Käytä ”Move Method” refaktorointia, ja siirrä logiikka Rental luokkaan
    - Käytä ”Replace Temp With Query”, jotta päästään eroon turhista muuttujista statement metodissa.
3. Lainauspisteiden laskemisessa ei myöskään käytetä Customer olion tietoja. Siirrä tämä logiikka oikeaan paikkaan ”Move Method” refaktoroinnilla
4. Statement metodissa lasketaan vuokrauksen kokonaishinta ja lainauspisteiden kokonaismäärät. Tämä selvästi rikkoo SRP:tä.
    - Siirrä kokonaishinnan ja pisteiden laskenta omiin metodeihin. (Extract Method)
    - Älä välitä siitä että, joudutaan käymään lainauslista läpi useampaan kertaan. Tämä voidaan myöhemmin optimoida, jos osoittautuu ongelmaksi.
5. Nyt näyttäisi olevan mahdollista tehdä uusi statementXml metodi, joka tulostaa kuitin xml-muotoon ja käyttää kuitin tietojen laskentaan samoja metodeja kuin statement.
    - Jos haluat, toteuta xml tulostus käyttäen ”Form Template Method” refaktorointia.

2. Ehtolauseen korvaaminen polymorfismilla

Uuden elokuvatyypin lisääminen näyttäisi aiheuttavan muutoksia switch-lauseeseen, jossa lasketaan elokuvan hinta, samoin tarvitaan myös muutoksia logiikaan, jossa lasketaan lainauspisteet. Tämä rikkoo selvästi SRP:tä!

1. Hinnan laskemiseen käytetään tietoa elokuvan tyypistä ja vuokrauksen kestosta. Näyttäisi siltä, että hinnan laskentalogiikka kuuluisi Movie luokkaan.
    - Siirrä laskentalogiikka Movie luokkaan, käyttämällä ”Move Method” refaktorointia.
2. Lainauspisteiden laskenta näyttäisi kuuluvan kanssa Movie luokkaan.
3. Nyt näyttäisi siltä, että Movie luokasta voidaan periä eri elokuvatyyppien kantaluokat, RegularMovie jne… ja toteuttaa niille oma logiikka lainauspisteiden ja hinnan laskemiseksi.

- Ongelma: Elokuvien tyyppi voi vaihtua ajan kuluessa, joten ei voidakaan periä elokuvatyyppejä suoraan Moviesta, koska luokkien tyyppi ei voi vaihtua.
- Ratkaisu: Käytetään ”Replace Type Code with State/Strategy” refaktorointia, ja lisätään Movie luokalle uusi Hinta jäsenolio.
- Hinta luokka on siis vastuussa elokuvan hinnoittelusta ja lainauspisteiden laskemisesta.
- Hinta luokasta voidaan periä eri hinnoittelulogiikan muodostavat luokat.



4. Käytä ”Self Encapsulate Field” refaktorointia, ja kapseloi Hinta jäsenolion asetus Movie luokkaan.
   - Huom! Elokuvan hinta asetetaan kokonaisluvulla luokan ulkopuolelta, joten tämä toiminnallisuus joudutaan säilyttämään, koska ei ole pääsyä ulkopuoliseen koodiin.
5. Käytä ”Move Method” refaktorointia ja siirrä hinnan laskeminen Moviesta Price olioon.
6. Toteuta hinnan laskenta Price luokan aliluokissa ja korvaa switch-lauseke yksi ehto kerrallaan.
7. Kun Price luokan aliluokat vastaavat hinnan laskennasta, muuta Price luokan laskentametodi abstraktiksi.
8. Siirrä lainauspisteiden laskenta samalla tavoin Price luokkaan. Jätä tällä kertaa Price luokkaan oletustapa laskea lainauspisteet.
   If-lauseessahan on oletushaara!
9. Nyt on siis mahdollista lisätä uusi hinnoitteluluokka klassikoille.

Liitetiedostot:

- RefactoringExcerciseJava.zip
- RefactoringExcerciseNET.zip