1 Design Patterns

Markera om följande påståenden är sanna eller falska:
(+1 för rätt svar, ingen förändring för fel svar)

En Observable är en klass med data som andra klasser kan vara intresserade av

- ○ Sant ✔
- ○ Falskt

Objektet main:GUIController, som är en Controller, ansvarar för att kontrollera att användaren använder gränssnittet rätt.

- ○ Sant
- ○ Falskt ✔

Objektet main:GUIController, som är en Controller, ansvarar för att skicka vidare händelser som användaren genererar mot gränssnittet till andra delar av applikationen som utför själva jobbet.

- ○ Sant ✔
- ○ Falskt

Ett Strategy pattern består av minst tre klasser med rollerna Context, AbstractStrategy, och ConcreteStrategy

- ○ Sant ✔
- ○ Falskt

Designmönstret Factory handlar om att all data (Facts) skall samlas i så få klasser som möjligt.

- ○ Sant
- ○ Falskt ✔

Totalpoäng: 5

2 Design Patterns

Markera om följande påståenden är sanna eller falska:
(+1p för rätt svar, ingen förändring för fel svar)

En Factory är ansvarig för att skapa objekt från rätt klasser.

- ○ Sant ✔
- ○ Falskt

Stategy Pattern handlar om att man skall ha en strategi för att fördela ansvar mellan olika klasser.

- ○ Sant
- ○ Falskt ✔

I Observer pattern har man en klass som publicerar händelser, och en eller fler klasser som konsumerar händelser.

- ○ Sant ✔
- ○ Falskt

Singleton är ett sätt att skydda ditt program från förändringar i gränssnitt, t.ex. på inköpta komponenter.

- ○ Sant
- ○ Falskt ✔

Totalpoäng: 4

3 GRASP och Design Patterns

Markera om följande påståenden är sanna eller falska:
(+1 för rätt svar, ingen förändring för fel svar)

Controllers kan anropa andra Controllers.

- ○ Sant ✔
- ○ Falskt

Det får bara finnas en instans av en Information Expert.

- ○ Sant
- ○ Falskt ✔

En Controller är ansvarig för att ta emot systemhändelser och delegera till andra klasser för att genomföra den önskade operationen.

- ○ Sant ✔
- ○ Falskt

Enligt High Cohesion skall man se till att varje klass skall ha så få olika ansvarsområden som möjligt.

- ○ Sant ✔
- ○ Falskt

Singleton betyder att man bara får anropa klassen en gång

- ○ Sant
- ○ Falskt ✔

Strategy pattern använder sig av polymorfism

- ○ Sant ✔
- ○ Falskt

Abstract Factory används för att skapa rätt typ av objekt givet ett visst kontext, där resten av systemet inte behöver veta exakt vilken typ objektet är.

- ○ Sant ✔
- ○ Falskt

När man använder Layered så har man alltid ett GUI-lager, ett logik-lager, och ett data-lager.

- ○ Sant
- ○ Falskt ✔

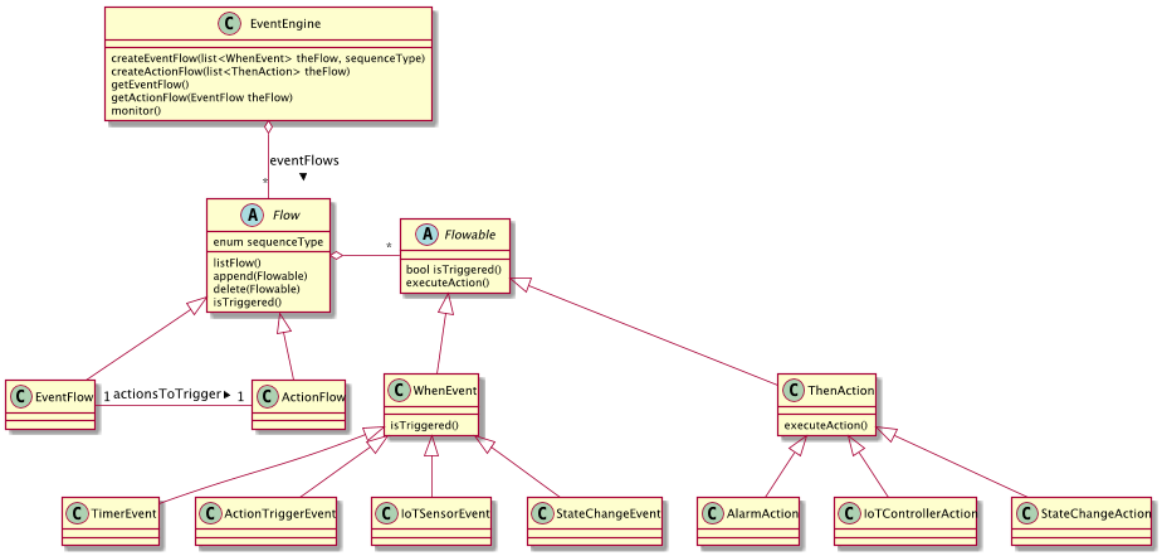
Totalpoäng: 8

4 Klassdiagram

Bakgrund

WhenThen är ett system där man bygger upp automatiserade flöden i sitt hem. Grunden är att man har en samling händelser (WhenEvents) och när dessa inträffar sker en eller flera handlingar (ThenActions). Flera händelser kan behövas (AND, OR, AND-IN-SEQUENCE) för att sätta igång arbetsflödet. Arbetsflödet kan i sin tur bestå av en eller flera handlingar.

En del av klassdiagrammet för WhenThen ser ut som nedan:



Markera om följande påståenden stöds (sant) av detta klassdiagram eller inte (falskt):

main:EventEngine kan innehålla oändligt många objekt av typen Flow.

- Sant ✔
- Falskt

te:TimerEvent är ett slags Flowable

- Sant ✔
- Falskt

af:ActionFlow kan bara tillhöra ett EventFlow i taget

- Sant ✔
- Falskt

Det går att skapa generiska WhenEvent som inte är instanser av någon av subklasserna till WhenEvent.

- Sant ✔
- Falskt

se:IoTSensorEvent tillhör både fireAlarm:Flow och doorbellRinging:Flow

- Sant
- Falskt ✔

ActionFlow innehåller den onödiga variablen sequenceType, vilket ger högre cohesion

- Sant
- Falskt ✔

Flow är en Controller för ett visst flöde.

- Sant ✔
- Falskt

Flowable är en Controller för klasserna WhenEvent och ThenAction

- Sant
- Falskt ✔

TimerEvent är en Information Expert på allt som har med en Timer-händelse att göra.

- Sant ✔
- Falskt

WhenEvent är en Information Expert på allt generellt som har att göra med en händelse, utom det som är specifikt för en viss typ av händelse.

- ○ Sant ✔
- ○ Falskt

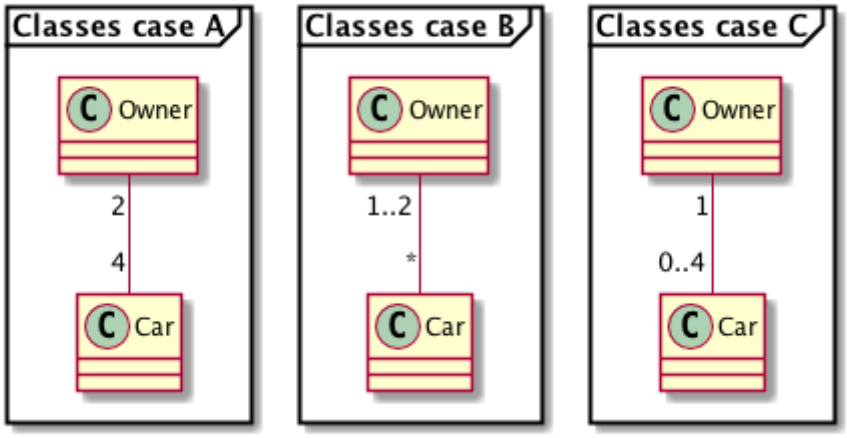
Klasserna Flow, Flowable, WhenEvent (inklusive subklasser), och ThenEvent (inklusive subklasser) är delar av designmönstret Strategy Pattern.

- ○ Sant ✔
- ○ Falskt

Totalpoäng: 11

5 Relationer mellan Klasser

Betrakta följande tre olika fall (Case A, Case B, och Case C) med olika relationer mellan klasserna Owner och Car:



Nedan anges sedan olika konstellationer av objekt. Markera vilket eller vilka klassdiagram (case A, case B, och/eller case C) som stödjer var och en av dessa konstellationer.

anna:Owner äger ingen bil.

- ○ Enbart Case C
- ○ Case B och C ✔
- ○ Case A och B
- ○ Enbart Case B

bengt:Owner äger fyra bilar.

- ○ Enbart Case C
- ○ Case B och C ✔
- ○ Case A och B
- ○ Enbart Case B

cissi:Owner och david:Owner äger tillsammans bilen a:Car.

- Enbart Case B ✔
- Case A och C
- Case B och C
- Enbart case C

emma:Owner och fredrik:Owner driver tillsammans en taxifirma där de har fyra bilar.

- Enbart case B
- Case A och B ✔
- Case B och C
- Enbart case C

gunilla:Owner och helge:Owner äger tillsammans två bilar.

- Case A och C
- enbart Case B ✔
- enbart case A
- Case B och C

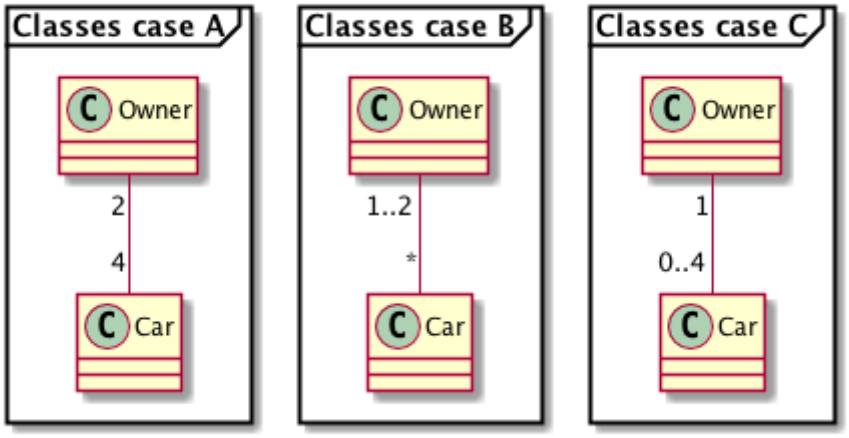
ingrid:Owner äger också två bilar.

- Case A och C
- enbart Case C
- Case B och C ✔
- Enbart case B

Totalpoäng: 6

6 Klasser och Implementation

Betrakta de tre fallen (Case A, Case B, och Case C) med olika relationer mellan klasserna Owner och Car:



Markera om följande påståenden är sanna eller falska:

I case A får det högst finnas 8 bilar i systemet

- ○ Sant
- ○ Falskt ✔

case A implementeras lämpligast med en array med pekare till objekt av typen Car i klassen Owner.

- ○ Sant ✔
- ○ Falskt

case B implementeras lämpligast med en array med pekare till objekt av typen Car i klassen Owner.

- ○ Sant
- ○ Falskt ✔

I case A måste Car också ha pekare tillbaka till Owner för att det skall fungera.

- ○ Sant
- ○ Falskt ✔

Enligt case A, om det finns tio objekt av typen Owner i systemet så måste det finnas minst 20 objekt av typen Car.

- ○ Sant
- ○ Falskt ✔

case B måste implementeras genom att klassen Car har två pekare till objekt av typen Owner.

- ○ Sant
- ○ Falskt ✔

I inget av fallen är det möjligt att byta ägare (Owner) till en bil (Car).

- ○ Sant
- ○ Falskt ✔

Totalpoäng: 7

ℹ Betygsgränser

Betygsgränserna för denna tenta är:

Betyg	Procent	Poäng
MAX	100%	41
A	90%	37
B	80%	33
C	70%	29
D	65%	26
E	60%	24

Lycka till!