S03 - Calculator

Von Burkhard Hampl
4AHITT

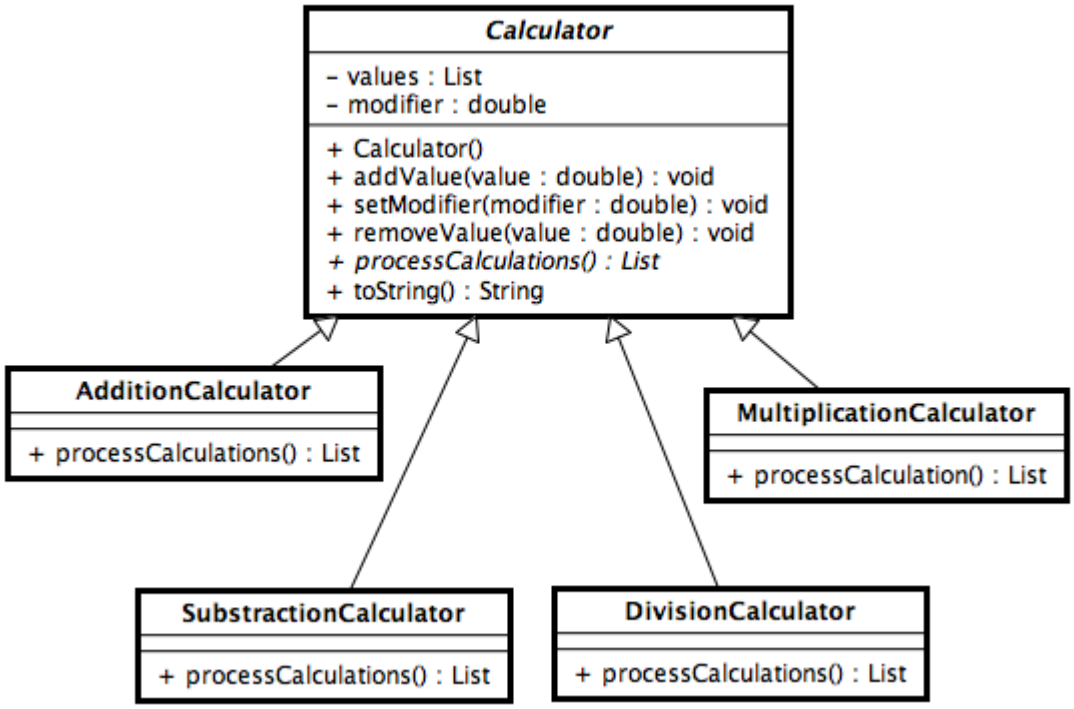
12.11.2014

Implementierung des Strategy-Pattern

Inhaltsangabe

Aufgabenstellung

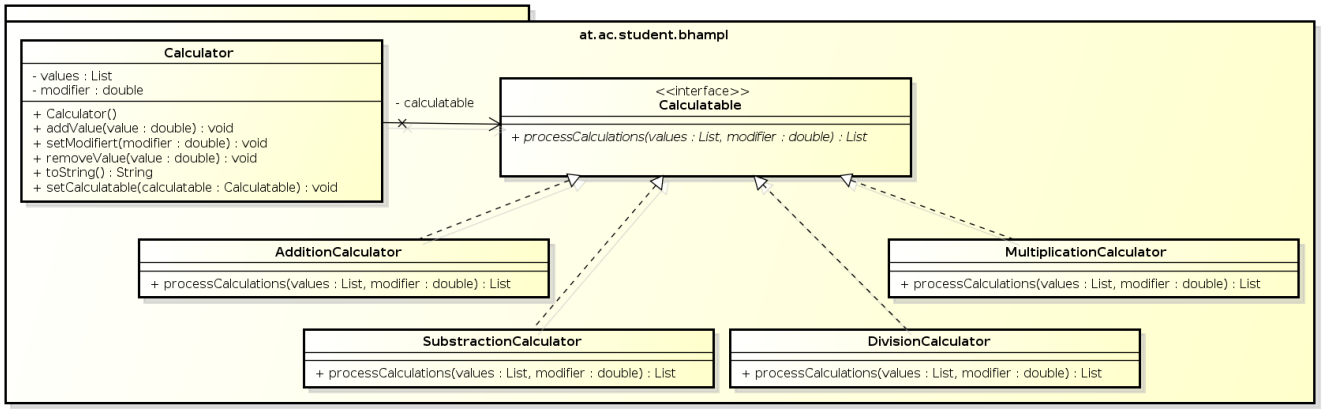
Ändere folgendes UML-Diagramm so um, dass es dem Strategy-Pattern entspricht und implementiere dann dieses.



Die abstrakte Klasse Calculator hat die Aufgabe, Werte aus einer Liste mit einem modifier zu verändern und das Ergebnis als neue Liste zurück zu geben. Dazu dient die abstrakte Methode processCalculations, die in den konkreten Subklassen so überschrieben wurde, dass sie je nach Klasse die Werte aus der Liste mit dem modifier addiert, subtrahiert, multipliziert oder dividiert.

Designüberlegung

Da es bei dieser Aufgabe es um das Strategy-Pattern geht, werde ich deise auch verwenden und implementieren.



Ich habe bei den Designüberlegung, darauf geachtet, dass ich den ausgangs „Code“, so gut wie möglich, gleich behalte und nur dort wo es wirklich nicht möglich war, habe ich was verändert. Dabei bin ich drauf gekommen, dass ich es nicht so lösen würde, wenn ich diese Aufgabe ganz frei gemacht hätte.

Leider habe ich bei den Überlegungen zu *processCalculations* die Parameter vergessen, so konnte man der Methode keine werte übergeben. Diesen Fehler, habe ich leider erst bei der Implementation bemerkt.

Arbeitsaufteilung

Da es sich um eine Einzelaufgabe handelt mach ich alles.

Aufwandsabschätzung

Am Anfang der Aufgabe habe ich für die ganze Aufgabe 2 Stunden berechnet

Endzeitaufteilung

Aufgabe	Zeit in h
UML-Umbau	0.5
Git und Eclipse einrichten	0.3
Implementation	0.5
Testen	0.75

Dokumentation	0.5
Geamt	2.55

Arbeitsdurchführung

Als erstes habe ich in Astah das UML-Diagramm „geändert“, das fertige UML habe ich dann in Java exportiert und die JavaDoc Kommentare geschrieben. Danach bin ich in etwa nach „test-driven development“ vorgegangen, d.h. ich habe zuerst die Test geschrieben und erst dann die Implementation durchgeführt.

Resultate

- Ich habe das Strategy-Pattern angewendet
- Mal wieder ein bisschen Git (/GitHub) und Eclipse dazu gelernt

Niederlagen

- Bei der Designüberlegung hab ich nicht alles beachtet und so musste ich bei der Implementierungsphase noch was ändern.

Testbericht

Ich habe mir zuerst die Tests überlegt und erst dann implementiert, so waren zuerst alle Test fehlerhaft und erst nach und nach wurden sie „grün“. Dabei bin ich am Ende sogar fast auf die 100% code coverage gekommen.

Quellenangaben