

Softwaretechnik 1 - 4. Tutorium

Tutorium 03

Felix Bachmann | 26.06.2017

KIT - INSTITUT FÜR PROGRAMMSTRUKTUREN UND DATENORGANISATION (IPD)



- 1 Orga
- 2 Recap
- 3 Stellvertreter
- 4 Vermittler
- 5 Gruppenarbeit
- 6 Tipps

Ansage der Übungsleiter

- ab jetzt keine Abgabe per Mail mehr!
⇒ auch nicht in Ausnahmefällen

4. Übungsblatt Statistik

Allgemein



Aufgabe 1 (Zustandsdiagramm - LEZ): \emptyset



Aufgabe 2 (Abbottsche Methode): Ø



Aufgabe 3 (iMage-GUI): Ø



Aufgabe 4 (Geheimnisprinzip): Ø

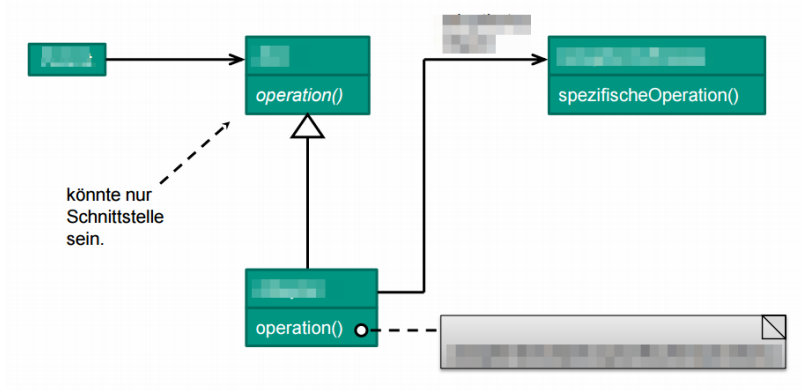


Was bisher geschah..

- haben uns erste Entkopplungsmuster angeschaut

Was bisher geschah..

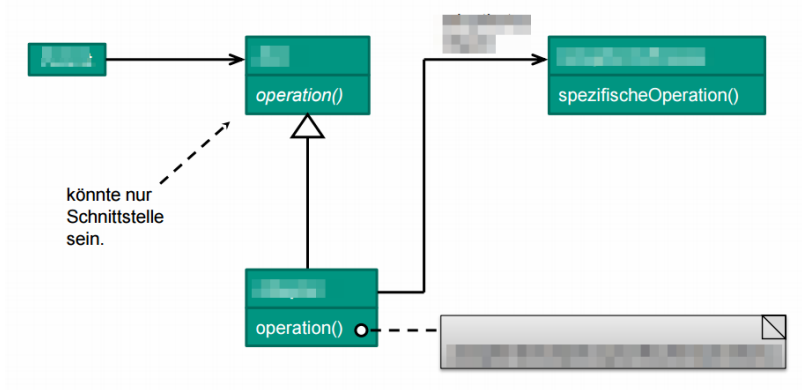
- haben uns erste Entkopplungsmuster angeschaut
⇒ Beobachter, Iterator, Adapter



Welches Entwurfsmuster?

Was bisher geschah..

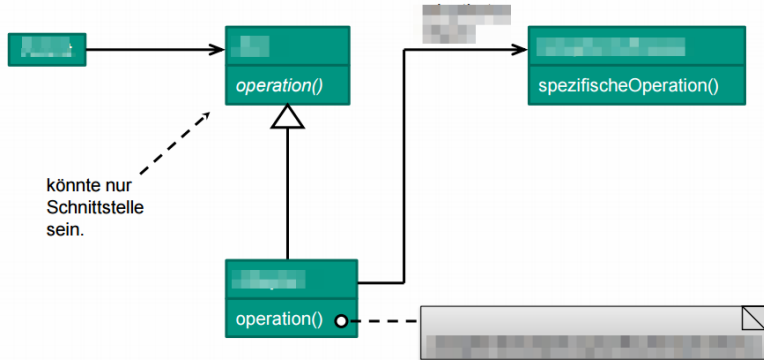
- haben uns erste Entkopplungsmuster angeschaut
⇒ Beobachter, Iterator, Adapter



Welches Entwurfsmuster? (Objekt-)Adapter

Was bisher geschah..

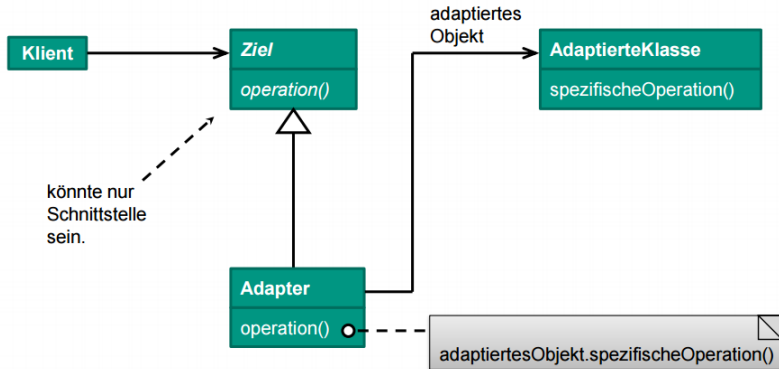
- haben uns erste Entkopplungsmuster angeschaut
⇒ Beobachter, Iterator, Adapter



Welche Klassen?

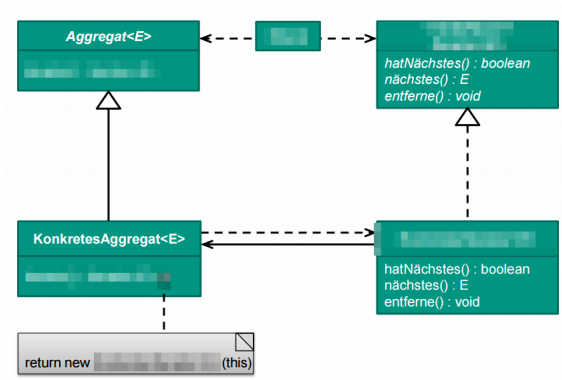
Was bisher geschah..

- haben uns erste Entkopplungsmuster angeschaut
 - ⇒ Beobachter, Iterator, Adapter



Was bisher geschah..

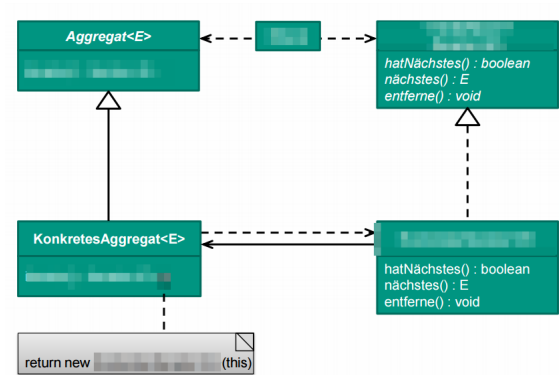
- haben uns erste Entkopplungsmuster angeschaut
⇒ Beobachter, Iterator, Adapter



Welches Entwurfsmuster?

Was bisher geschah..

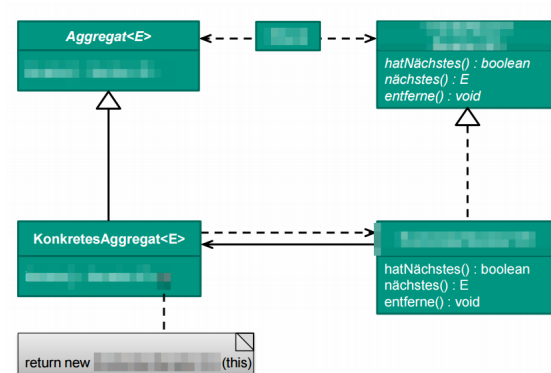
- haben uns erste Entkopplungsmuster angeschaut
⇒ Beobachter, Iterator, Adapter



Welches Entwurfsmuster? Iterator

Was bisher geschah..

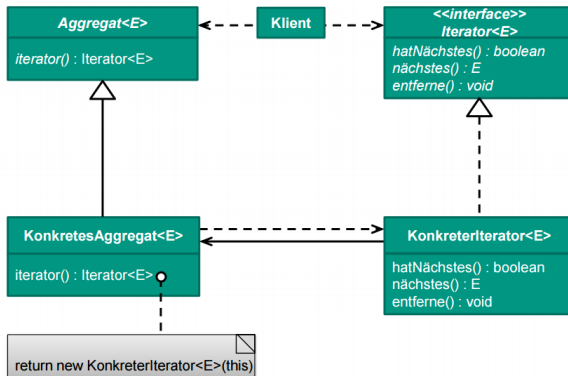
- haben uns erste Entkopplungsmuster angeschaut
⇒ Beobachter, Iterator, Adapter



Welche Klassen und Methoden?

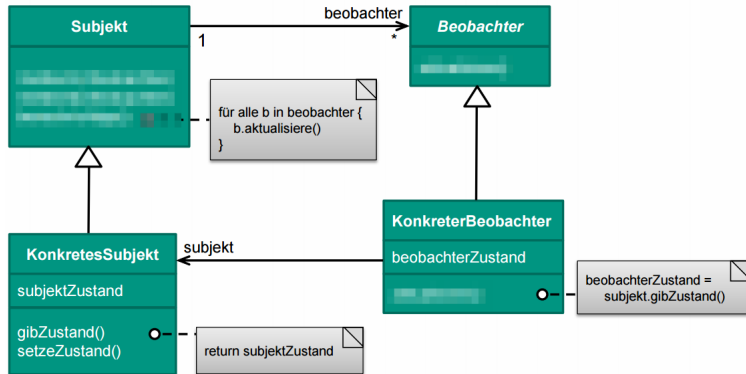
Was bisher geschah..

- haben uns erste Entkopplungsmuster angeschaut
⇒ Beobachter, Iterator, Adapter



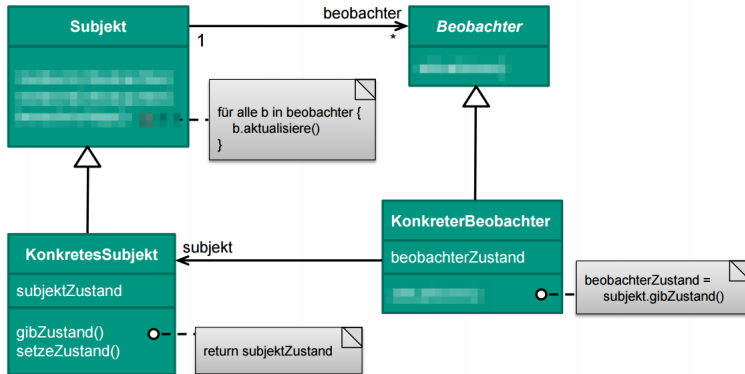
Was bisher geschah..

- haben uns erste Entkopplungsmuster angeschaut
⇒ Beobachter, Iterator, Adapter



Was bisher geschah..

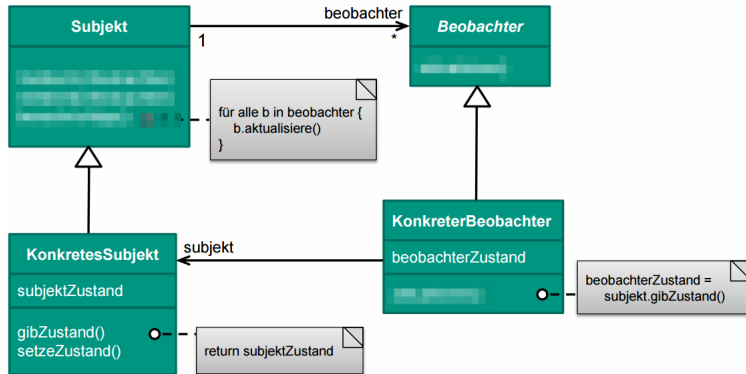
- haben uns erste Entkopplungsmuster angeschaut
⇒ Beobachter, Iterator, Adapter



Ist wohl ein Beobachter :)

Was bisher geschah..

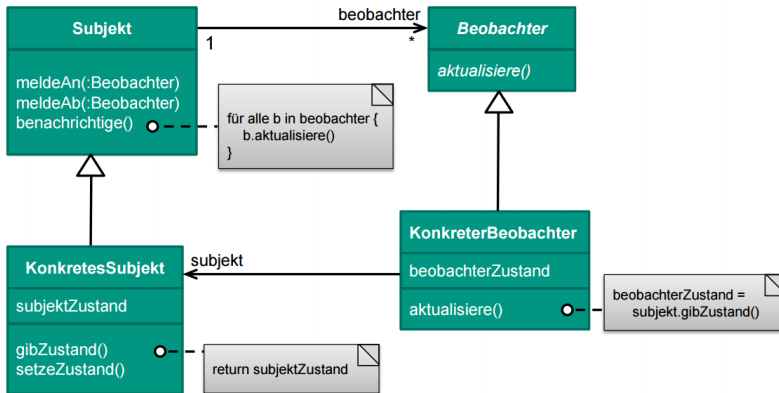
- haben uns erste Entkopplungsmuster angeschaut
⇒ Beobachter, Iterator, Adapter



Ist wohl ein Beobachter :) Klassen, Methoden?

Was bisher geschah..

- haben uns erste Entkopplungsmuster angeschaut
⇒ Beobachter, Iterator, Adapter



■ Entkopplungs-Muster

- Adapter fertig
- Beobachter fertig
- Iterator fertig
- **Stellvertreter**
- **Vermittler**
- (Brücke)

■ Varianten-Muster

■ Zustandshandhabungs-Muster

■ Steuerungs-Muster

■ Bequemlichkeits-Muster

Problem

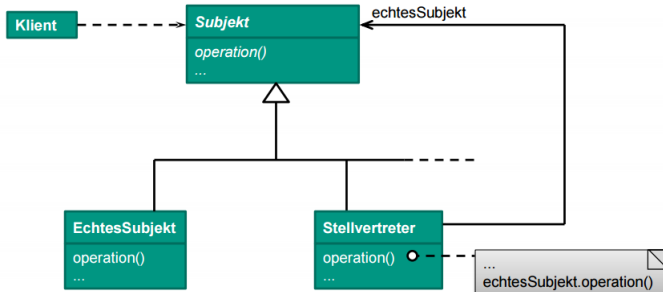
- wollen Zugriff auf ein Objekt kontrollieren, ohne seine Klasse zu ändern

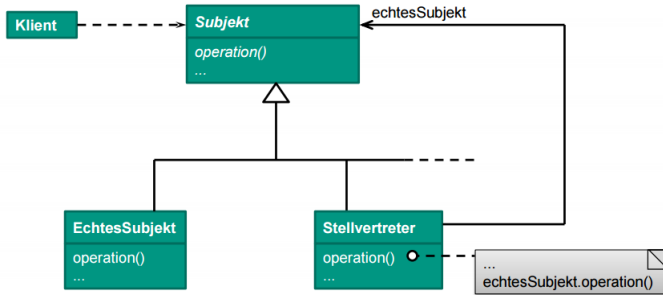
Problem

- wollen Zugriff auf ein Objekt kontrollieren, ohne seine Klasse zu ändern
⇒ Stellvertreter macht Zugriffskontrolle

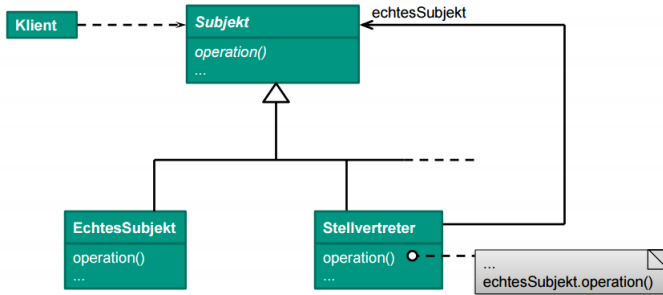
Problem

- wollen Zugriff auf ein Objekt kontrollieren, ohne seine Klasse zu ändern
⇒ Stellvertreter macht Zugriffskontrolle



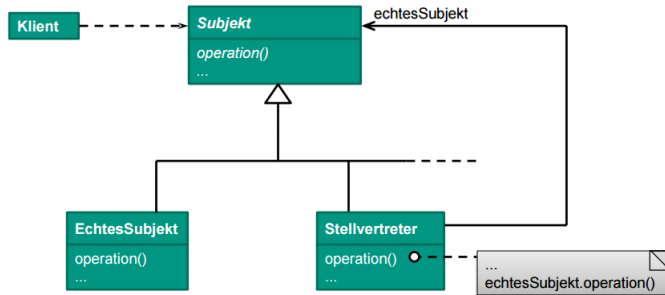


Entkopplung?



Entkopplung?

- Klient hat keinen direkten Zugriff auf das echte Subjekt



Entkopplung?

- Klient hat keinen direkten Zugriff auf das echte Subjekt
- Stellvertreter hat Relation zu Oberklasse (!), echtes Subjekt austauschbar

Problem

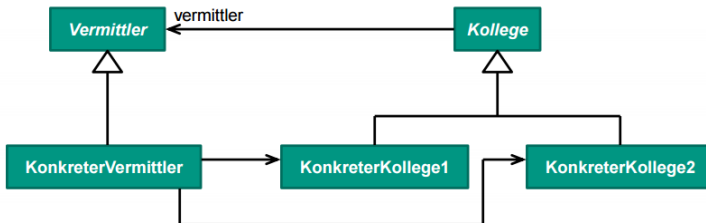
- mehrere voneinander abhängige Objekte

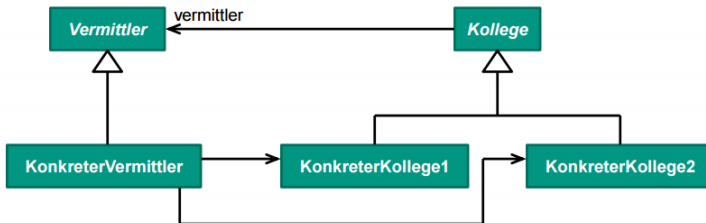
Problem

- mehrere voneinander abhängige Objekte
⇒ Zustände der Objekte von anderen Zuständen abhängig

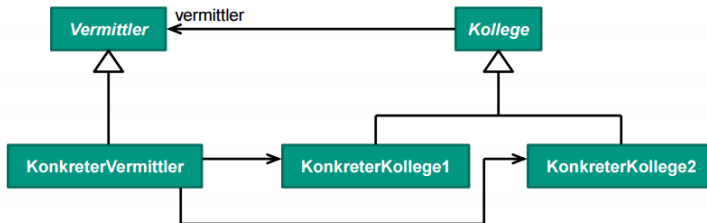
Problem

- mehrere voneinander abhängige Objekte
⇒ Zustände der Objekte von anderen Zuständen abhängig



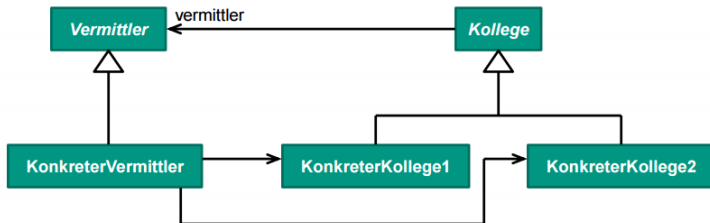


Entkopplung?



Entkopplung?

- Kollegen kennen sich nicht direkt



Entkopplung?

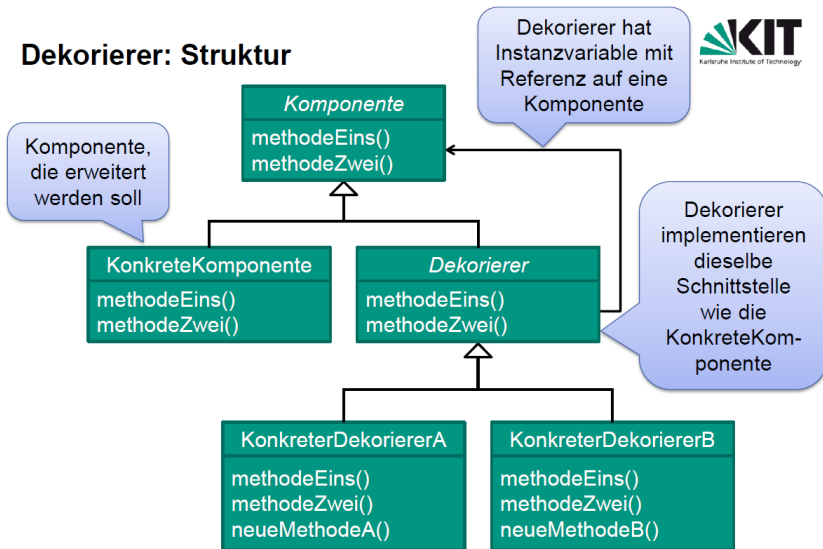
- Kollegen kennen sich nicht direkt
⇒ Hinzufügen eines Kollegen erfordert keine Änderung der alten Kollegen

- Entkopplungs-Muster fertig
- **Varianten-Muster**
 - (Abstrakte Fabrik)
 - (Besucher)
 - Schablonenmethode
 - Fabrikmethode
 - Kompositum
 - Strategie fertig
 - Dekorierer
- Zustandshandhabungs-Muster
- Steuerungs-Muster
- Bequemlichkeits-Muster

Jetzt: Gruppenarbeit

- 1 ihr kriegt pro Reihe eine Aufgabe
- 2 ihr habt Zeit zum Bearbeiten
- 3 ihr stellt den anderen vorne eure Lösung vor

Dekorierer: Struktur



Wo Gemeinsamkeiten?

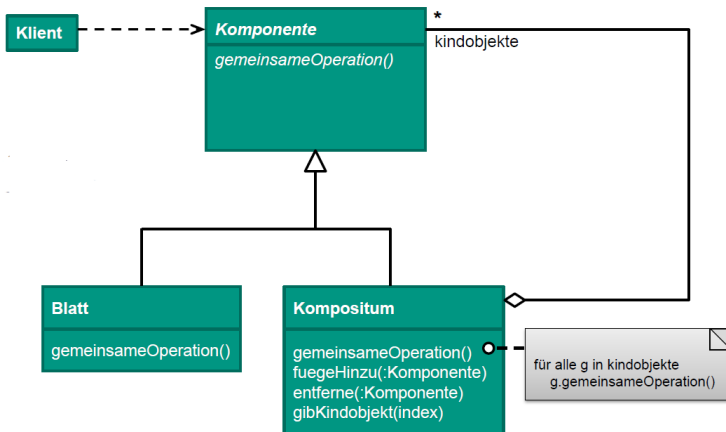
Die beiden Methoden `methodeEins()` und `methodeZwei()`.

Wo Variation?

In den KonkretenDekorierern bzw. ihren Methoden. Hier: `neueMethodeA()`, `neueMethodeB()`.

Wozu Instanzvariable?

Weiterleitung von Aufrufen der `methodeEins()` und `methodeZwei()` an die KonkreteKomponente.



Wo Gemeinsamkeiten?

gemeinsameOperation().

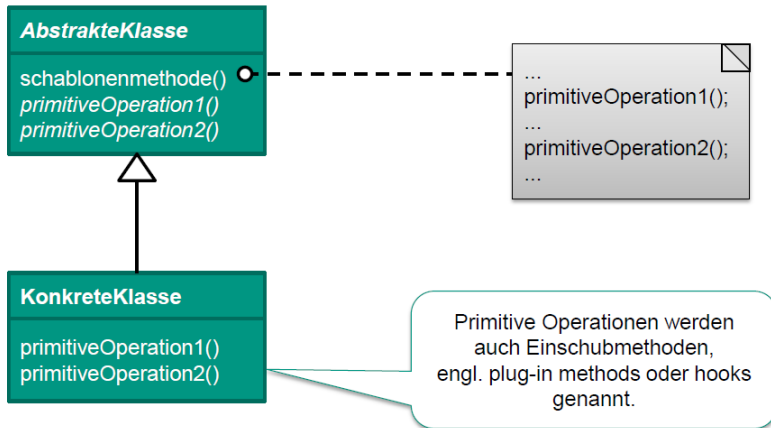
Wo Variation?

In Blatt/Kompositum-Klassen mit verschiedenen zusätzlichen Operationen.

Zusammengesetzt vs. nicht-zusammengesetzt

Kompositum = zusammengesetzt, Blatt = nicht-zusammengesetzt

Vorstellung Schablonenmethode



Wo Gemeinsamkeiten?

Reihenfolge der Methodenaufrufe in der Schablonenmethode.

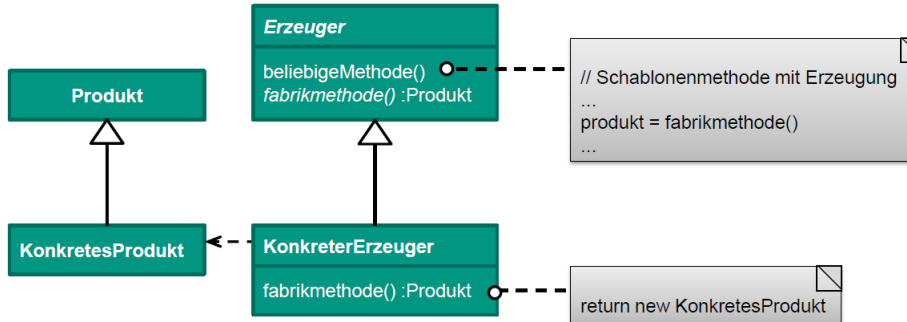
Wo Variation?

In den Einschubmethoden. (hier: `primitiveOperation1()` und `primitiveOperation2()`)

Schablonenmethode vs. Einschubmethode

Einschubmethode ist eine der Methoden, die von der Schablonenmethode aufgerufen wird und deren Implementierung in den Unterklassen stattfindet.

Vorstellung Fabrikmethode



Wo Gemeinsamkeiten?

Reihenfolge der Methodenaufrufe in der beliebigenMethode().

Wo Variation?

In der Fabrikmethode.

Klasse des Objekts, Oberklasse, Unterklasse

Klasse des Objekts = KonkretesProdukt, Oberklasse = Produkt,
Unterklasse = KonkreterErzeuger

Unterschied zu Schablonenmethode?

Fabrikmethode benutzen, wenn ein Objekt erzeugt wird. Fabrikmethode ist Einschubmethode des Musters "Schablonenmethode".

Wahr/falsch

Fabrikmethode ist eine Einschubmethode, keine Schablonenmethode.

Aufgabe 1: Manager-Deutsch und Architekturstile

- Architekturstile nochmal anschauen

Aufgabe 1: Manager-Deutsch und Architekturstile

- Architekturstile nochmal anschauen

Aufgabe 2: Iterator für Plug-Ins

- Iterator-Muster selbst benutzen

Aufgabe 3: Geometrify mit Entwurfsmustern

- überlegen, welches Entwurfsmuster **warum** Sinn macht

Aufgabe 3: Geomtrify mit Entwurfsmustern

- überlegen, welches Entwurfsmuster **warum** Sinn macht

Aufgabe 4: Geomtrify umstrukturieren

- Überlegungen aus Aufgabe 3 umsetzen

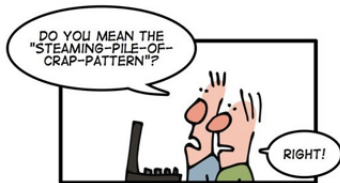
Aufgabe 5: GUI erweitern

- nochmal ServiceLoader \implies diesmal mit Primitiven

Abgabe

- Deadline am 5.7. um 12:00
- Aufgabe 1, 3 handschriftlich

Bis dann! (dann := 10.07.17)



THE HYPE IS LONG GONE BUT
DESIGN PATTERNS ARE STILL USEFUL