CAMPUS 02 GRAZ
FACHHOCHSCHULE DER WIRTSCHAFT

SOFTWARE DESIGN FOLIENSATZ 2

Decorator Pattern

Bernhard Fuchs
ZAM - WS 2025/26

TERMINE

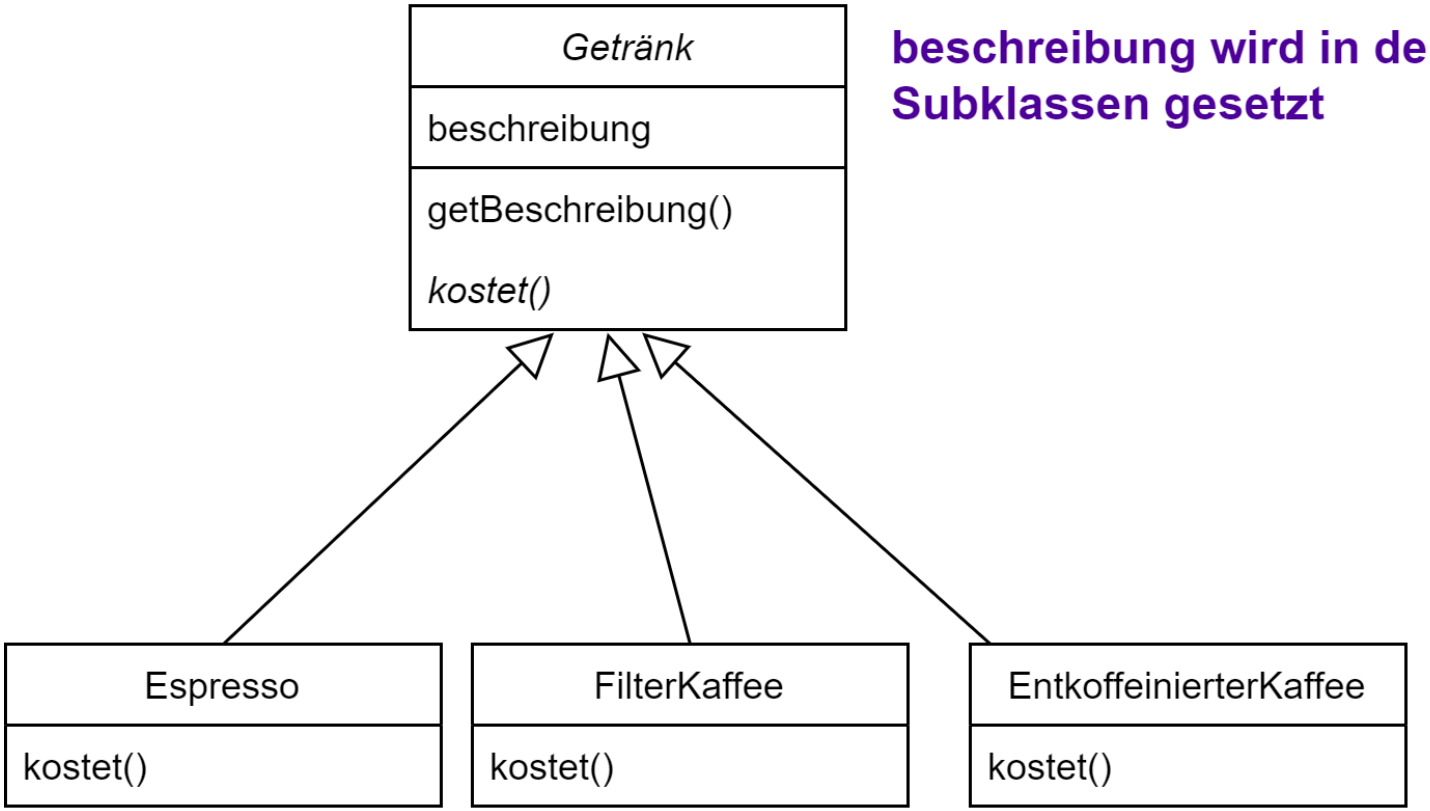
0004VD2005 SOFTWARE DESIGN (29,75UE IL, WS 2025/26)

Gruppe

	Tag	Datum	von	bis	Ort	Ereignis	Termintyp	Lerneinheit	Vortragende*r	Anmerkung
	Standardgruppe									
	Do	22.01.2026	08:15	11:45	CZ106	Abhaltung	fix	**1 Organisation + Strategy Pattern**		
Heute	Do	29.01.2026	08:15	11:45	CZ106	Abhaltung	fix	**2 Decorator Pattern + Singleton**		
	Do	05.02.2026	12:30	16:00	CZ106	Abhaltung	fix	**3 Observer**		
	Fr	06.02.2026	08:15	12:15	CZ106	Abhaltung	fix	**4 Factory**		
	Do	12.02.2026	08:15	13:00	CZ106	Abhaltung	fix	**5 Command + Adapter**		
	Fr	13.02.2026	08:15	12:15	CZ106	Abhaltung	fix	**6 Facade, Template, Iterator** mit Christian Hofer!		
	Do	19.02.2026	08:15	11:45	CZ106	Abhaltung	fix	**Wiederholung**		
	Fr	20.02.2026	08:15	12:15	CZ106	Prüfungstermin	fix	**Prüfung**		

DECORATOR PATTERN (WRAPPER)

KAFFEEHAUS - AUSGANGSSITUATION



beschreibung wird in den Subklassen gesetzt

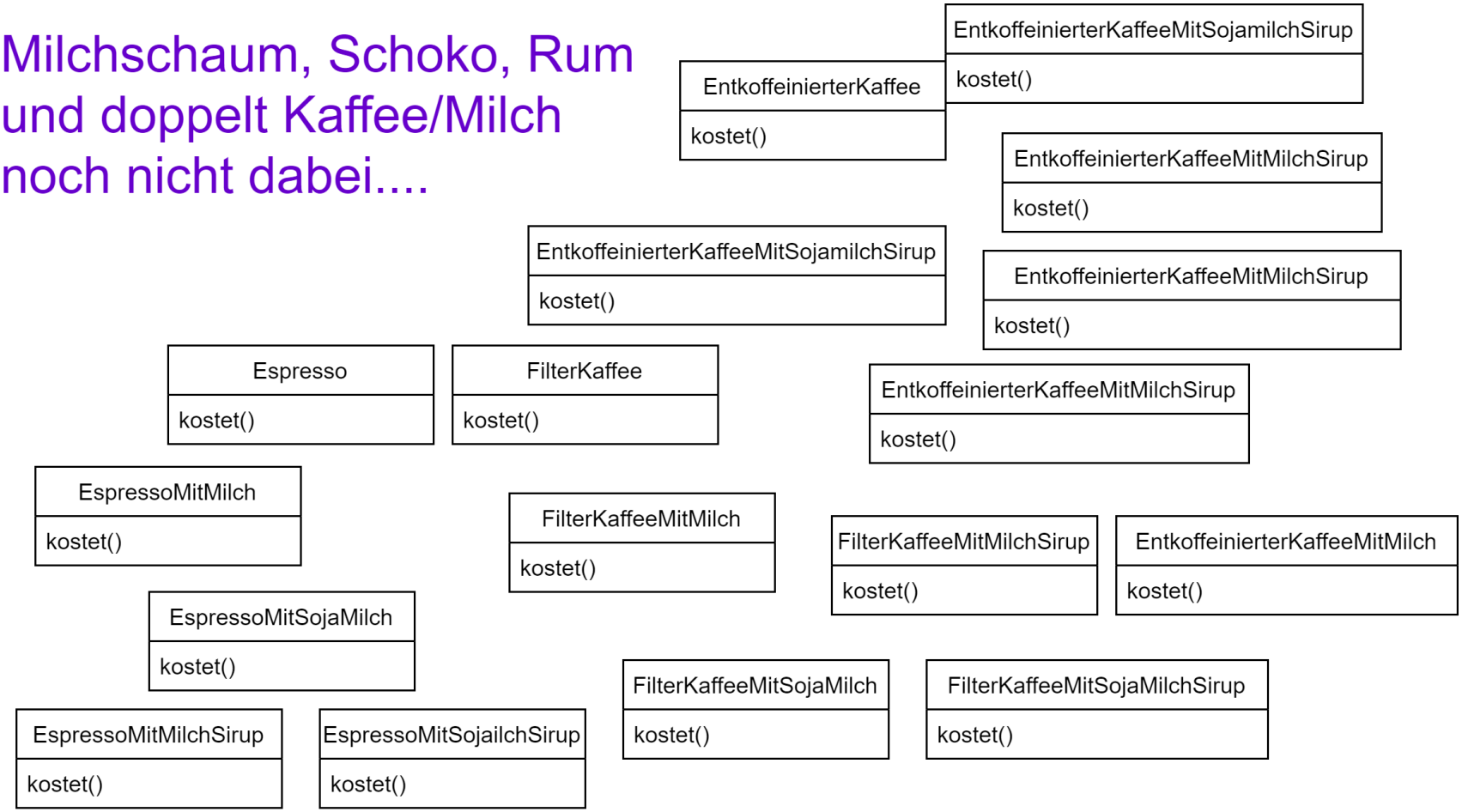
implementieren kostet()

DARF ES ETWAS MEHR SEIN?

- Milch
- Sojamilch
- Sirup
- Milchschaum
- Schokoeier

KAFFEE MIT MILCH, SOJAMILCH, MILCHSCHAUM, SIRUP, …

Milchschaum, Schoko, Rum und doppelt Kaffee/Milch noch nicht dabei....



HAB EINE IDEE…

Getränk
beschreibung
getBeschreibung() kostet() hatMilch() mitMilch() hatSoja() mitSoja() hatMilchschaum() mitMilchschaum()

kostet() wird hier implementiert

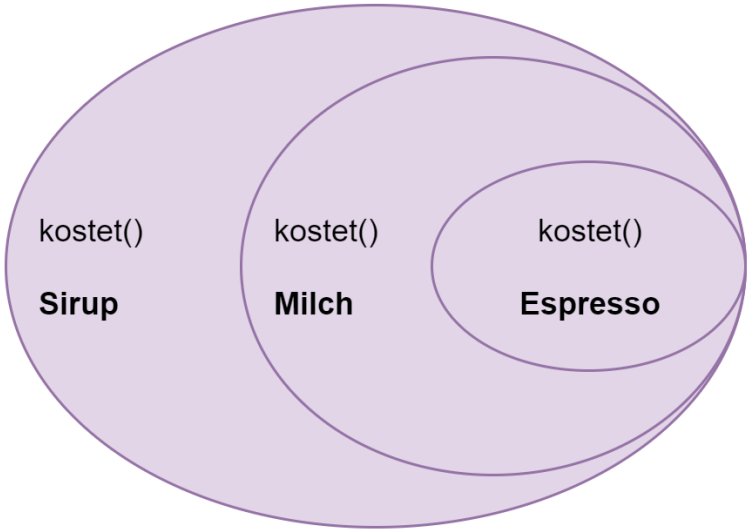
Berücksichtigt alle Sonderwünsche

OPEN-CLOSED DESIGN PRINZIP

Klassen sollen offen für Erweiterung, aber geschlossen für Veränderung sein.

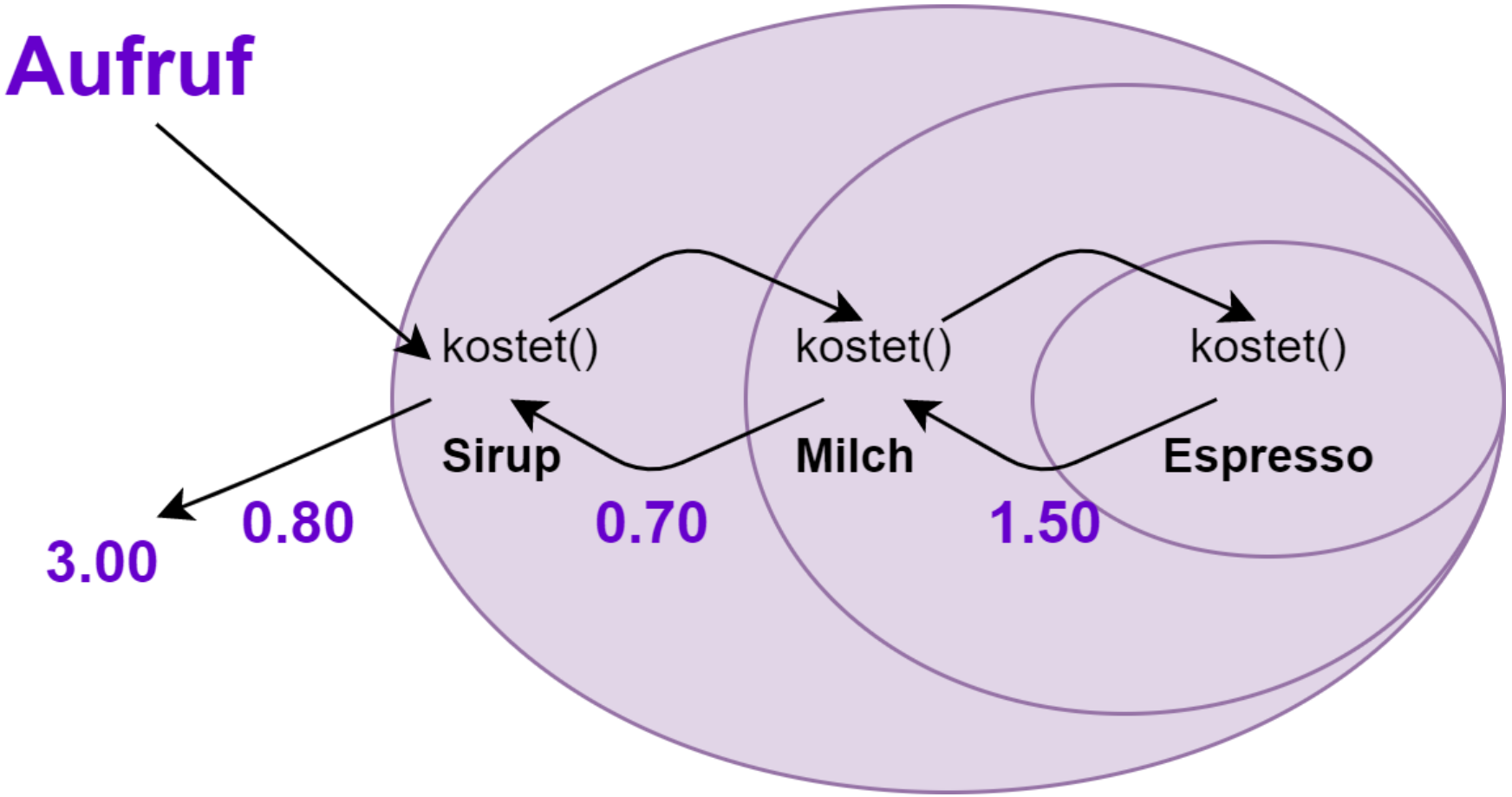
DECORATOR

Milch und Sirup sind Decorator
Spiegeln das Objekt das sie dekorieren



Haben gleiche Methoden und können wie Getränke behandelt werden

WIE BERECHNE ICH KOSTEN

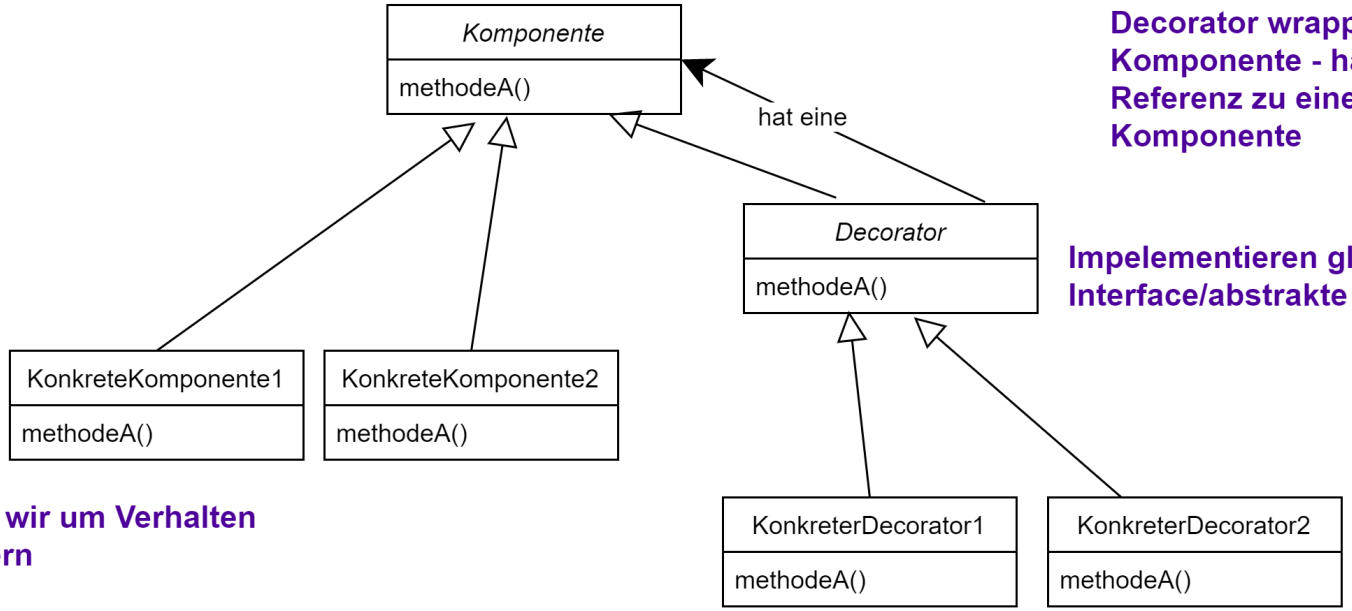


WIR WISSEN …

- Decorator haben gleichen Supertyp als Objekte die sie dekorieren.
- Mehrfache Verwendung möglich
- Decorator kann anstatt des Originals verwendet werden
- Decorator führt eigenes Verhalten kurz vor/nach Delegation an dekoriertes Objekt aus
- Dynamisch zu jedem Zeitpunkt möglich

WIE UMSETZEN?

Komponente und Decorator können Interface oder abstrakte Klasse sein



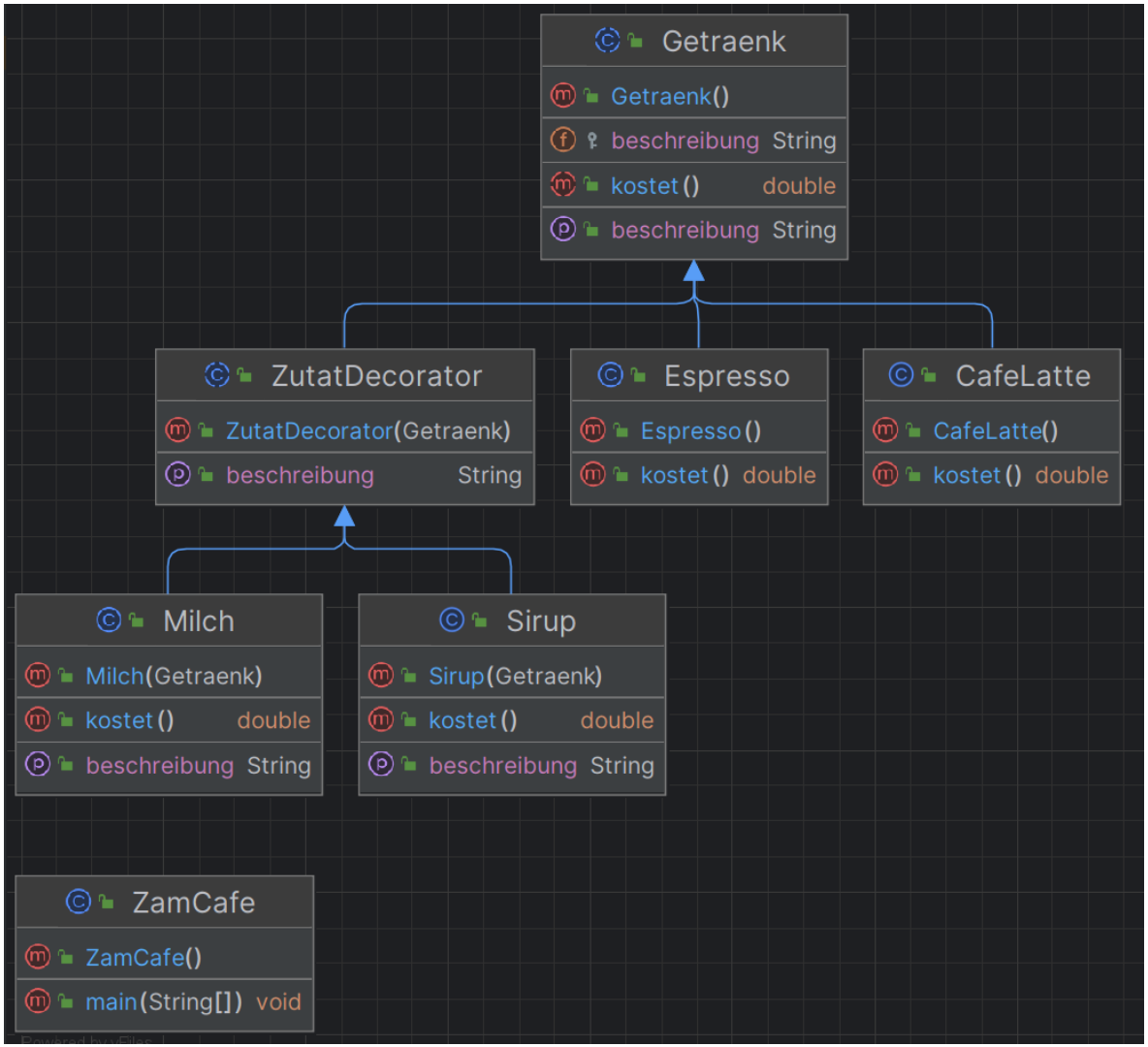
Decorator wrappt eine Komponente - hält eine Referenz zu einer Komponente

Impelementieren gleiches Interface/abstrakte Klasse

Wollen wir um Verhalten erweitern

Kann Komponente erweitern

Let‘s code


Getraenk
Getraenk()
beschreibung String
kostet() double
beschreibung String
ZutatDecorator
ZutatDecorator(Getraenk)
beschreibung String
Espresso
Espresso()
kostet() double
CafeLatte
CafeLatte()
kostet() double
Milch
Milch(Getraenk)
kostet() double
beschreibung String
Sirup
Sirup(Getraenk)
kostet() double
beschreibung String
ZamCafe
ZamCafe()
main(String[]) void

DECORATOR PATTERN

Decorator Pattern fügt dynamisch zusätzliche Verantwortung zu einem Objekt. Decorator sind flexible Alternativen zur Ableitung um Funktionalität zu erweitern.

ANWENDUNG: DECORATOR IN JAVA