MAYFLOWER

Strategy Pattern

Pattern DoJo, 02. März 2021

Definition

Das Strategy Pattern definiert
eine Familie austauschbarer Algorithmen
und kapselt diese in in Klassen,
die zur Laufzeit dynamisch geladen werden können.

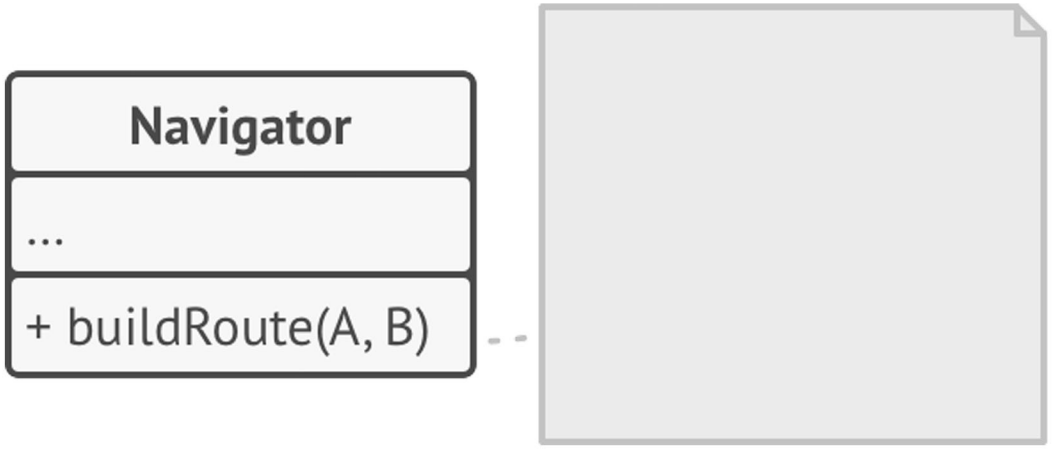
Verwendung

Unterschiedliche Verhaltensweisen und deren verwendete Algorithmen sind innerhalb einer Klasse fest integriert (z.B. durch Mehrfachverzweigungen), und sollen wiederverwendbar gemacht werden.

Verwandte Klassen unterscheiden sich in ihrem Verhalten unterschiedliche benötigen austauschbare Varianten eines Algorithmus. Hierdurch wird die betroffene Klasse flexibler gestaltet.

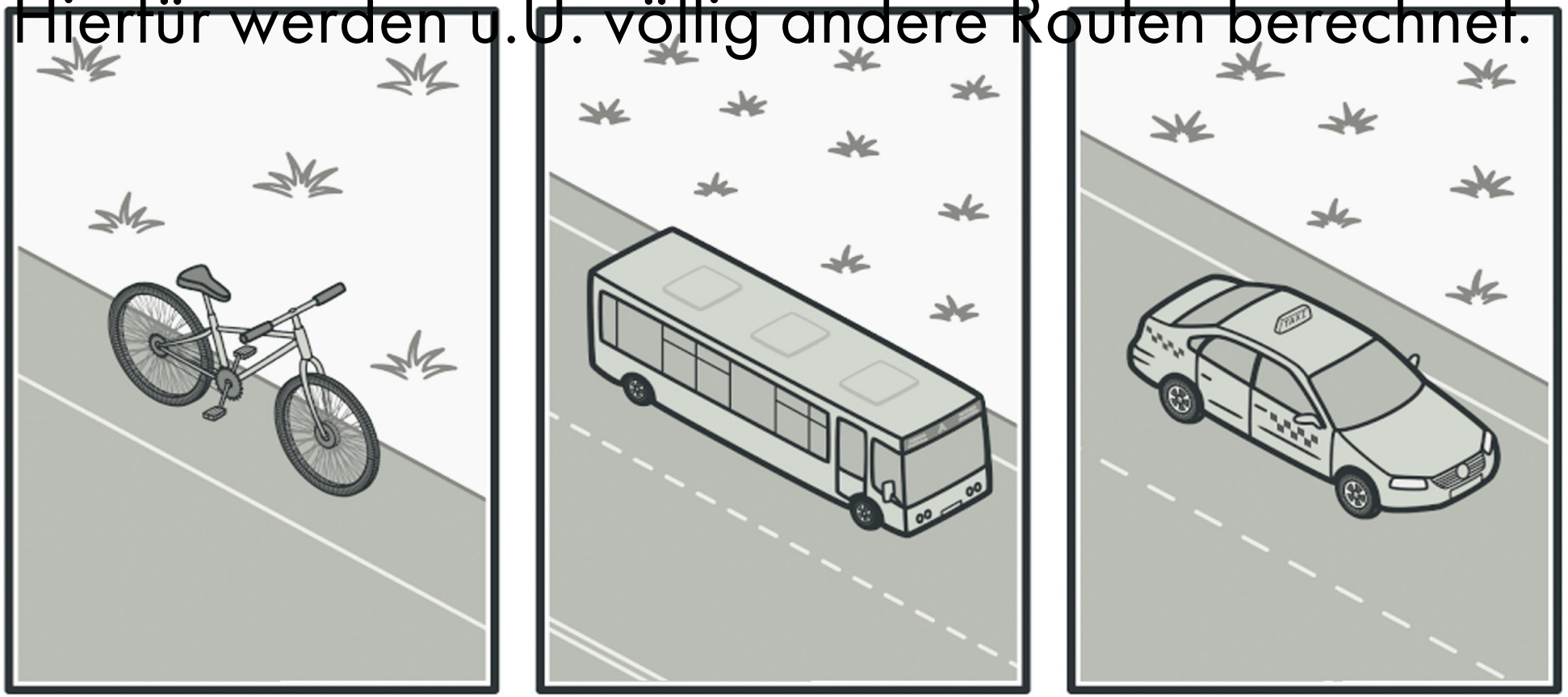
Problem

In der Navigator.buildRoute(A, B) wird die kürzeste Straßennavigation für PKW zwischen den beiden Punkten A und B berechnet.



Problem

nun soll das Programm auch die kürzeste Route für Fahrradfahrer und den ÖPNV berechnen können. Hierfür werden u.U. völlig andere Routen berechnet.

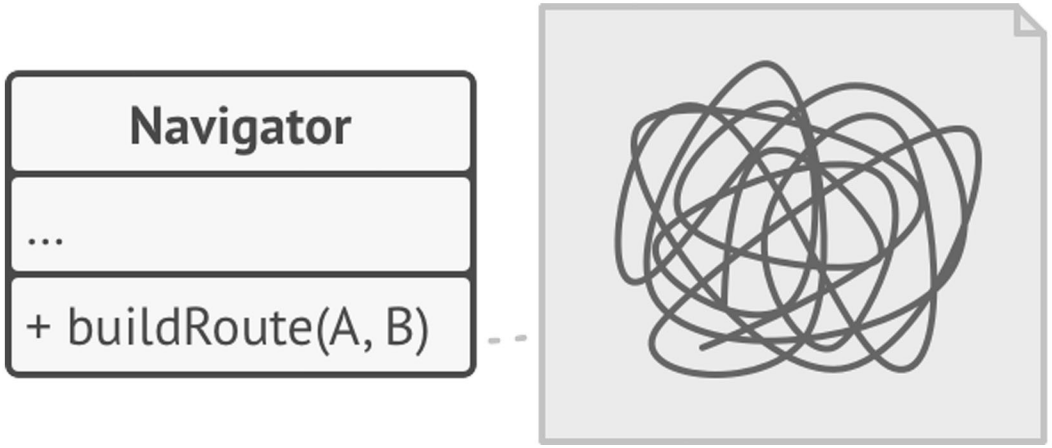


Problem

Diese würde man in der Navigator.buildRoute(A, B)
mit switch-Funktionen
und dem Aufruf unterschiedlicher Funktionen lösen.

Problem

Auf kurz oder lang würde diese Erweiterung die Klasse Navigator aufblähen und viel Code innerhalb einer Einheit definieren, der sauber getrennt werden sollte.



Implementierung

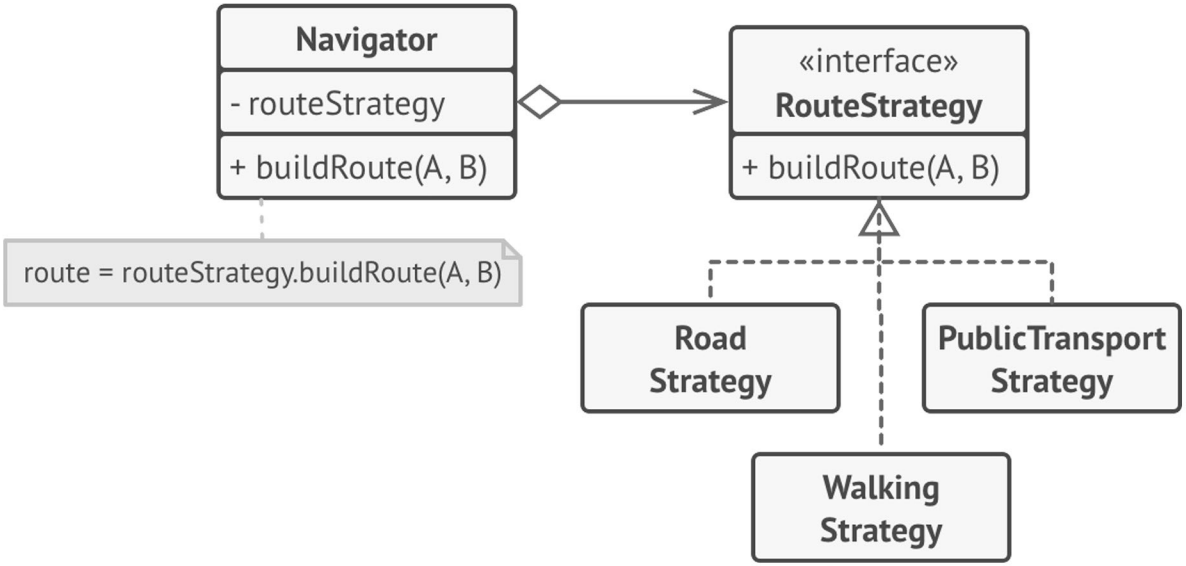
In the context class, identify an algorithm
that's prone to frequent changes.
It may also be a massive conditional that selects
and executes a variant of the same algorithm at runtime.
Declare the strategy interface
common to all variants of the algorithm.
One by one, extract all algorithms into their own classes.
They should all implement the strategy interface.
In the context class, add a field for storing a reference
to a strategy object.
Provide a setter for replacing values of that field

Anwendbarkeit

Kontextgebunden - Es geht darum dass die Logik/Verhalten kontextunabhängig ausgelagert wird!
(Also nicht code in konkrete methode einer subklasse umzieht!)
Use the Strategy pattern when you want to use different variants of an algorithm within an object and be able to switch from one algorithm to another during runtime.
The Strategy pattern lets you indirectly alter the object's behavior at runtime by associating it with different sub-objects which can perform specific sub-tasks in different ways.
Use the Strategy when you have a lot of similar classes that

Lösung

Die buildRoute(A, B) wird von einer neue Schnittstelle RouteStrategy abstrakt deklariert. Für alle benötigten Varianten werden Klassen erstellt welche die Schnittstelle implementieren.



Vorteile

You can swap algorithms used inside an object at runtime. You can isolate the implementation details of an algorithm from the code that uses it. You can replace inheritance mposition. Open/Closed Principle. You can introduce new str without having to change the context.

Es wird eine Familie von Algorithmen definiert.

Es wird die Auswahl aus verschiedenen Implementierungen ermöglicht und dadurch erhöhen sich die Flexibilität und die Wiederverwendbarkeit. Es können Mehrfachverzweigungen vermieden werden und dies erhöht die Übersicht des Codes. egien bieten eine Alternative zur Unterklassenbildung der Kont

Nachteile

f you only have a couple of algorithms and they rarely change,
here's no real reason to overcomplicate the program with new
classes and interfaces that come along with the pattern.
Clients must be aware of the differences between
strategies to be able to select a proper one.
A lot of modern programming languages have functional
type support that lets you implement different
versions of an algorithm inside a set of anonymous functions.
use these functions exactly as you'd have used the strategy obj
bloating your code with extra classes and interfaces.