Design Patterns

Problem

How not to create a bike?

Design Patterns

What’s a design pattern?

- is typical solution to commonly problems
- is general concept for solving problems
- is high-level description of a solution to problems

What does the pattern consist of?

- Intent
- Motivation
- Structure
- Code example

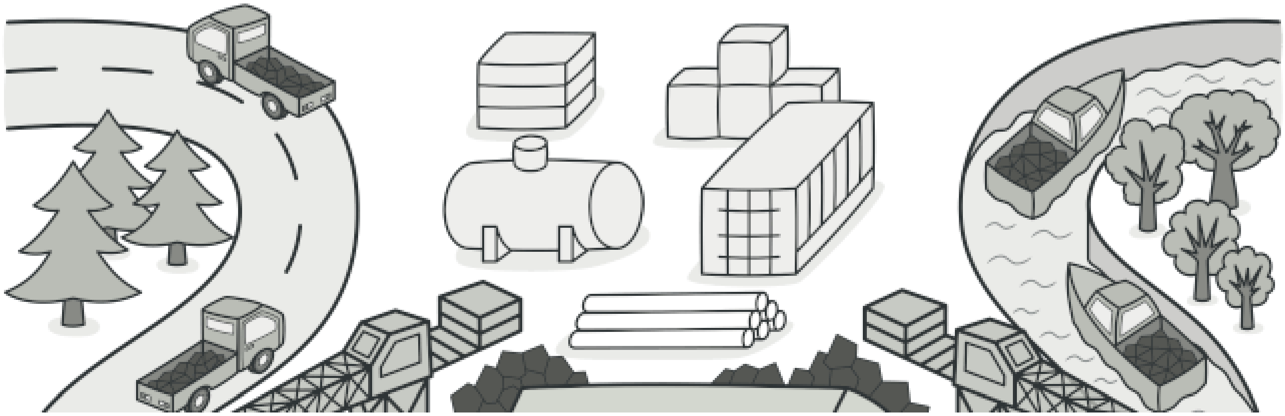
Classification of patterns

- Creational patterns
- Structural patterns
- Behavioral patterns

Creational patterns

Factory Method

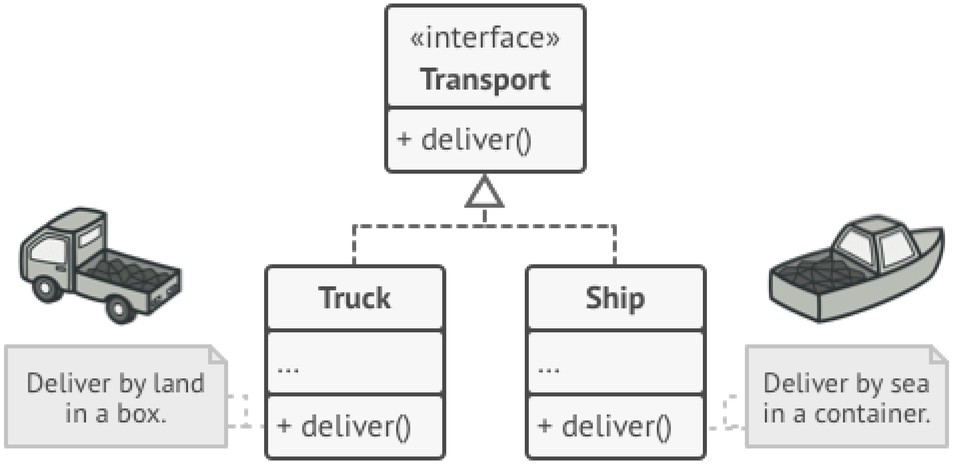
Intent



Motivation: problem

Motivation: solution

Motivation: solution



Motivation: solution

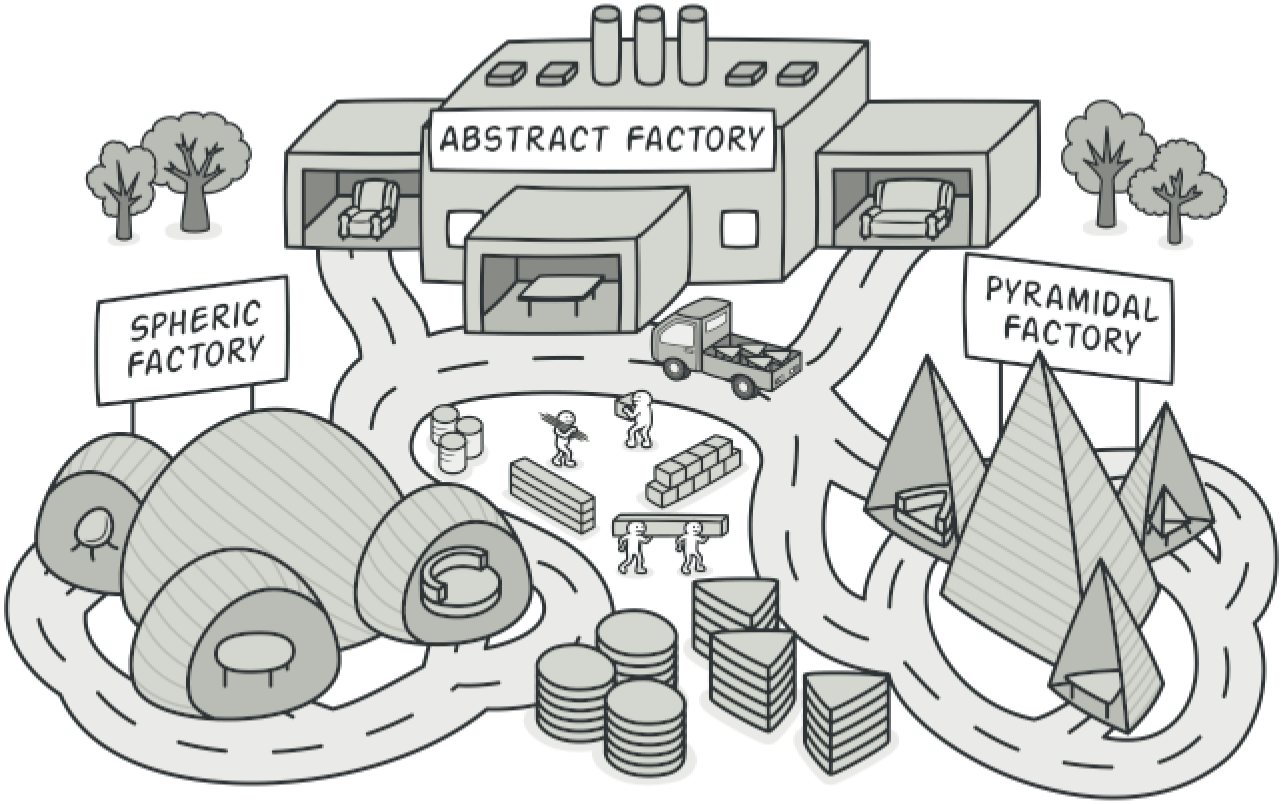
Structure

Structure

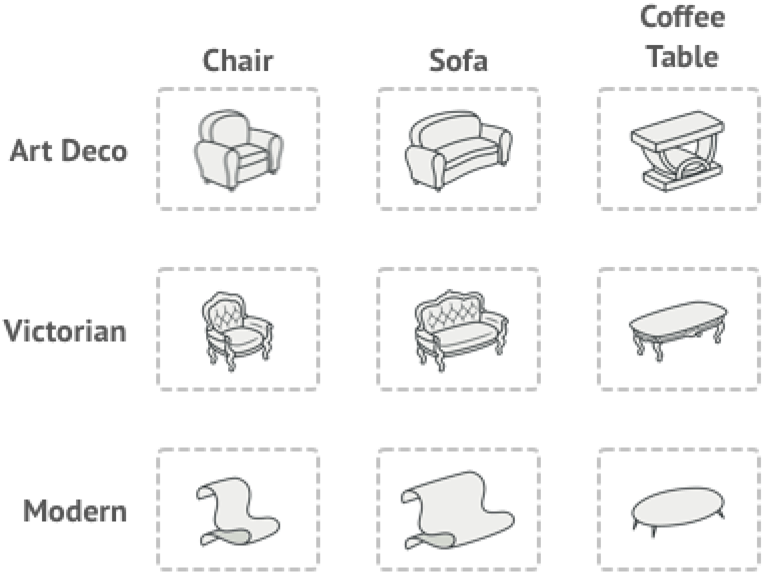
Example

Abstract Factory

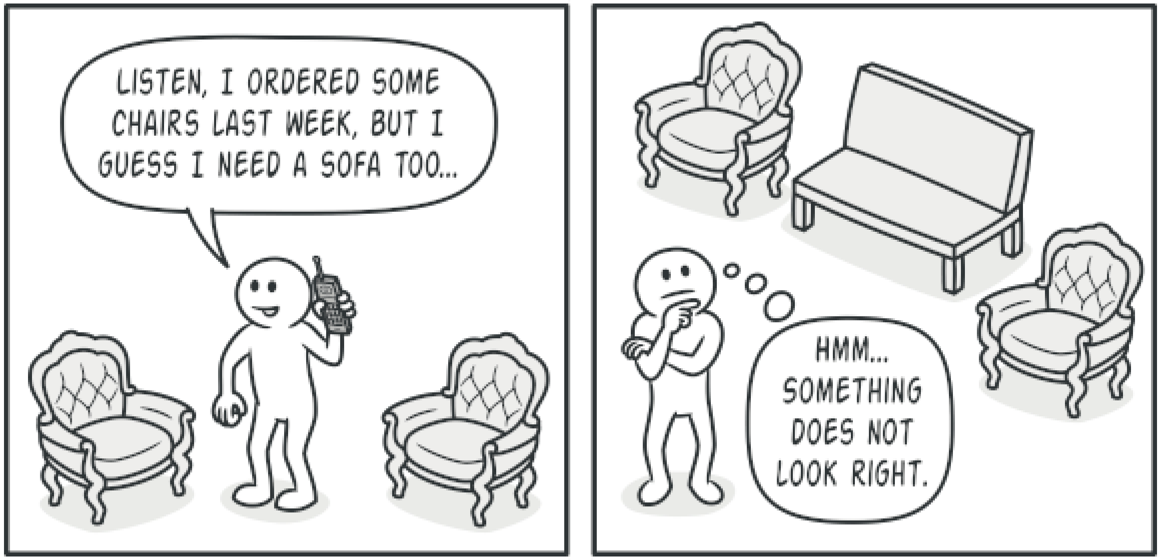
Intent



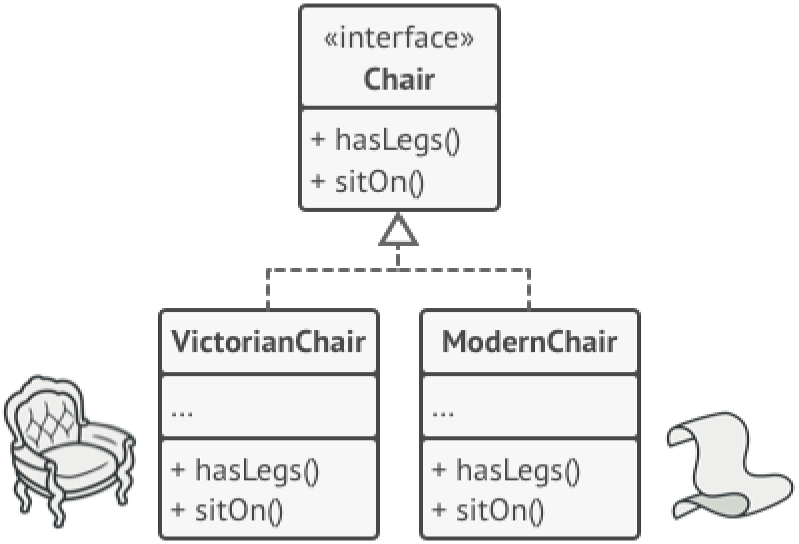
Motivation: problem



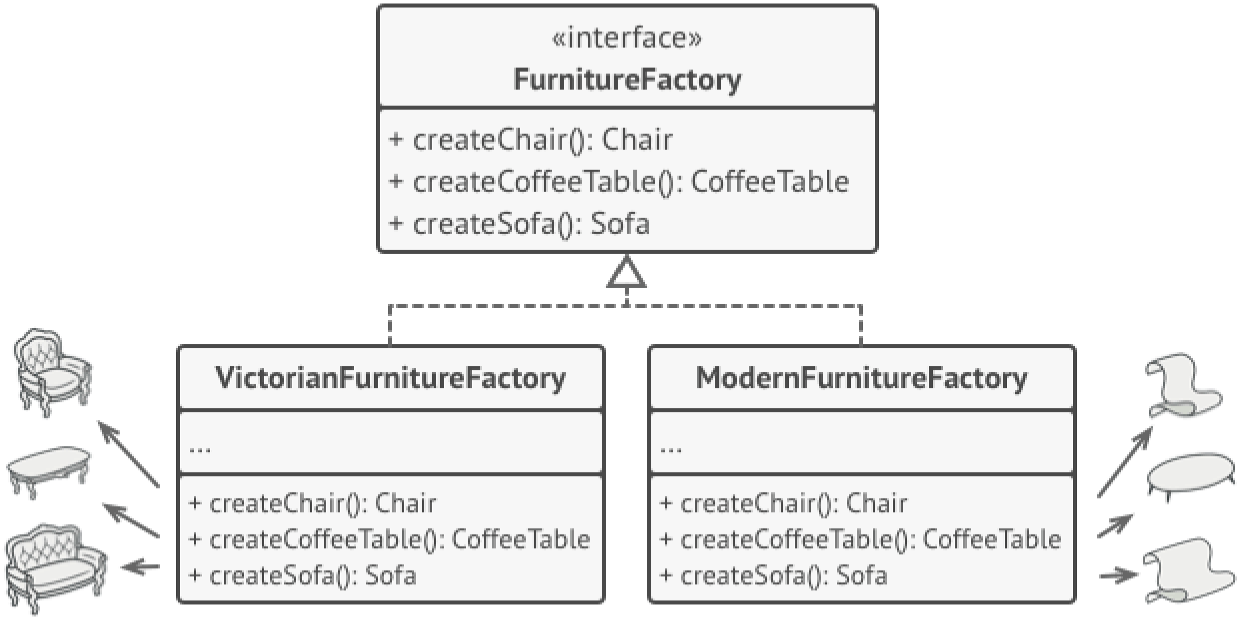
Motivation: problem


LISTEN, I ORDERED SOME CHAIRS LAST WEEK, BUT I GUESS I NEED A SOFA TOO...
HMM... SOMETHING DOES NOT LOOK RIGHT.

Motivation: solution



Motivation: solution



«interface»
FurnitureFactory

+ createChair(): Chair
+ createCoffeeTable(): CoffeeTable
+ createSofa(): Sofa

VictorianFurnitureFactory

...

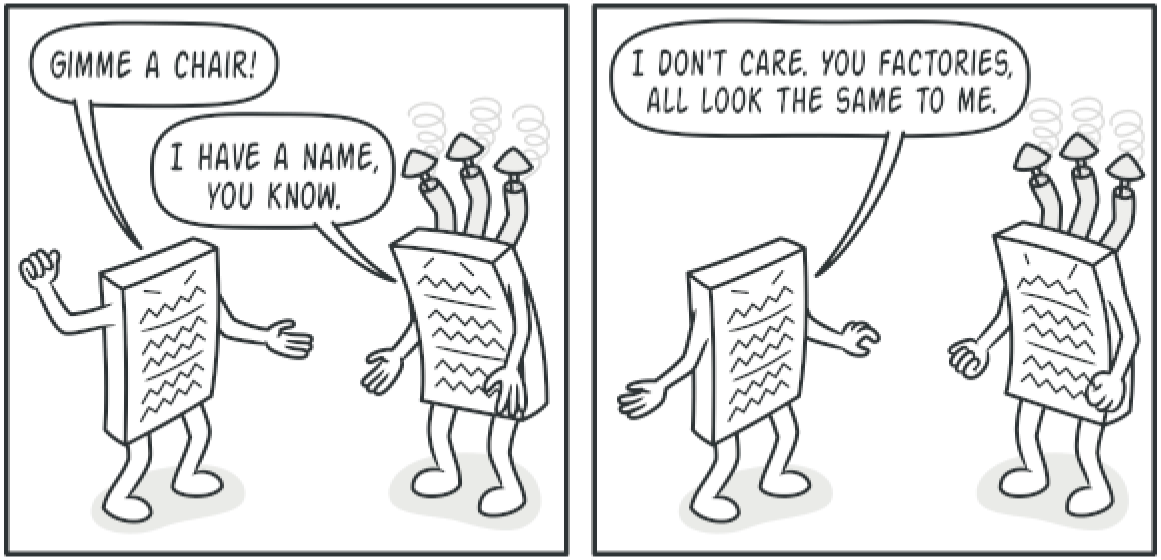
+ createChair(): Chair
+ createCoffeeTable(): CoffeeTable
+ createSofa(): Sofa

ModernFurnitureFactory

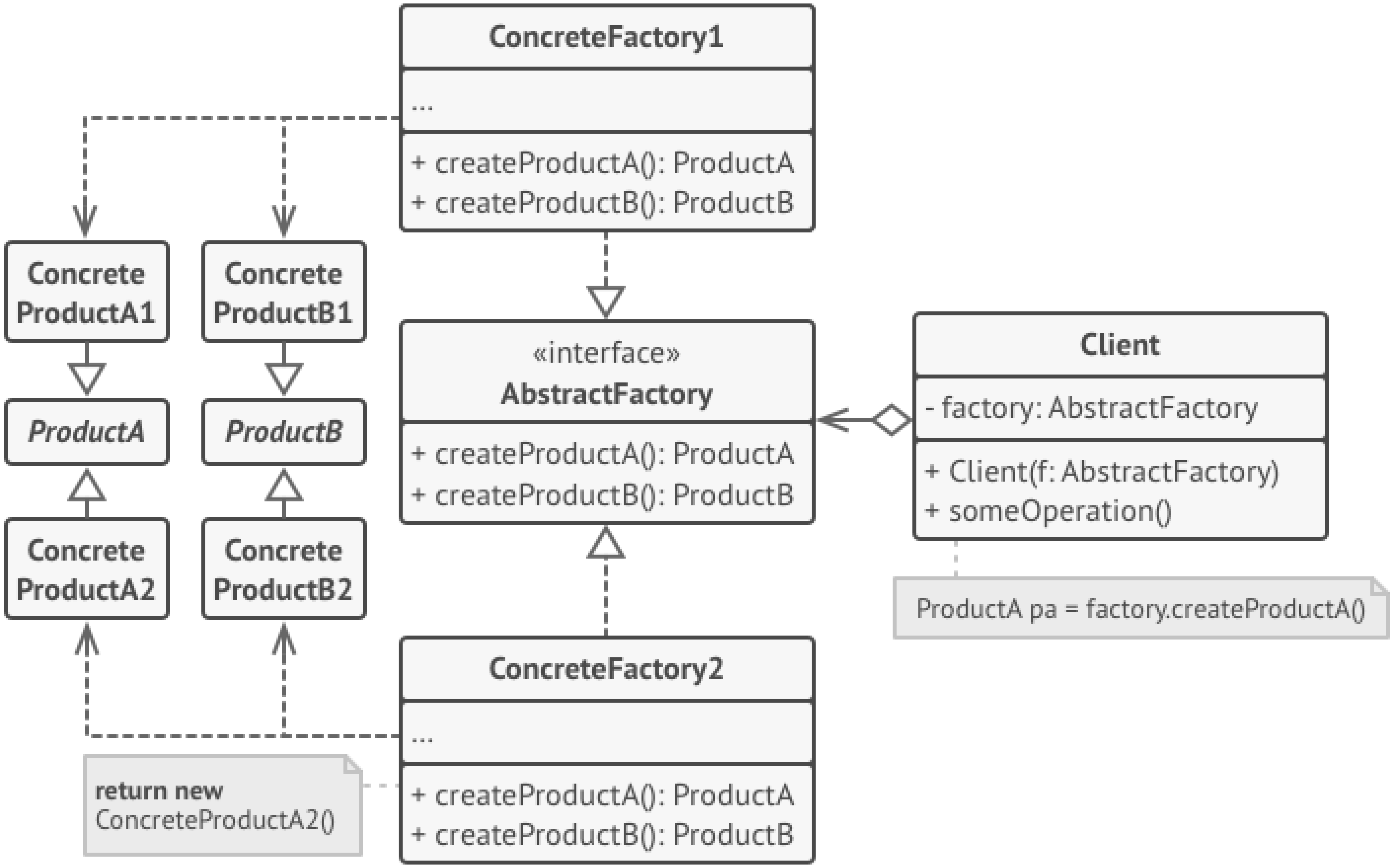
...

+ createChair(): Chair
+ createCoffeeTable(): CoffeeTable
+ createSofa(): Sofa

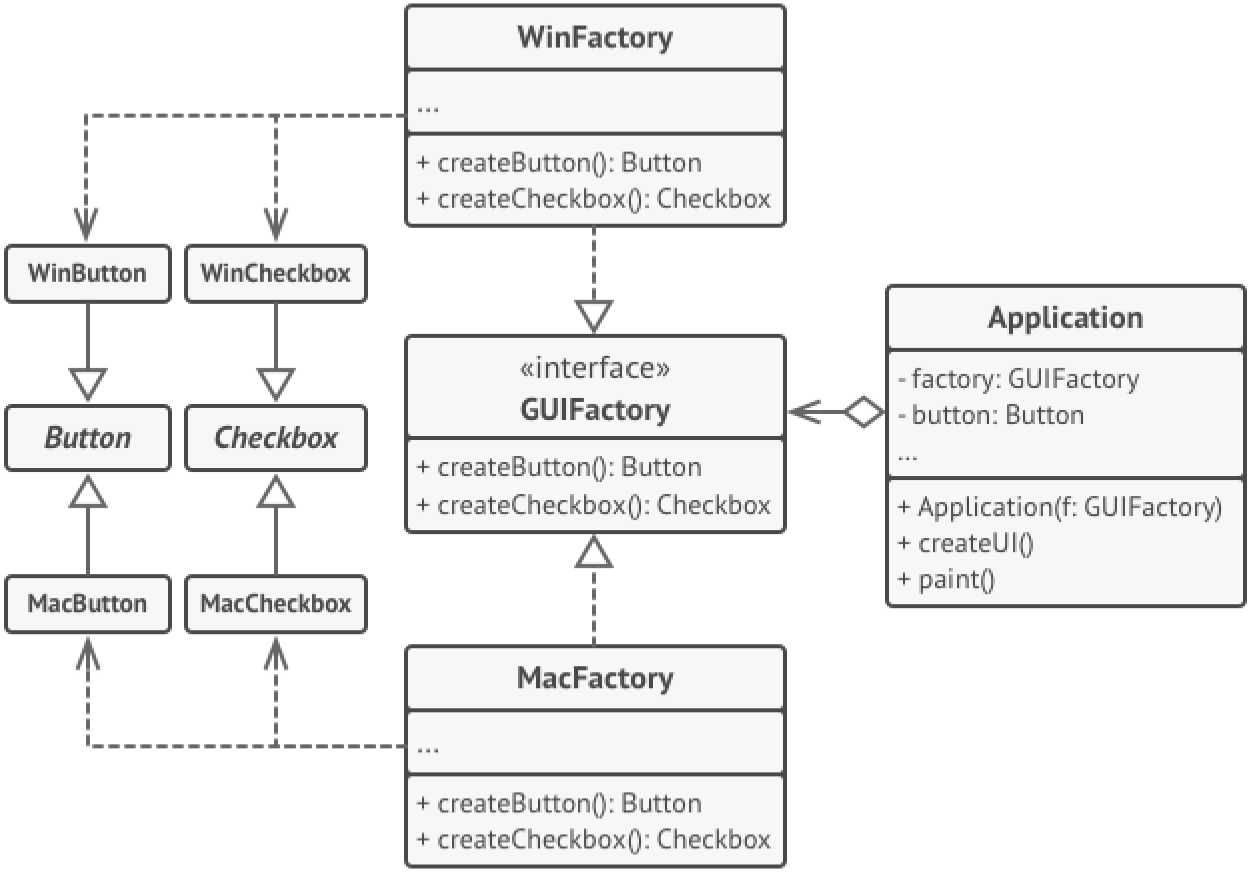
Motivation: solution



Structure

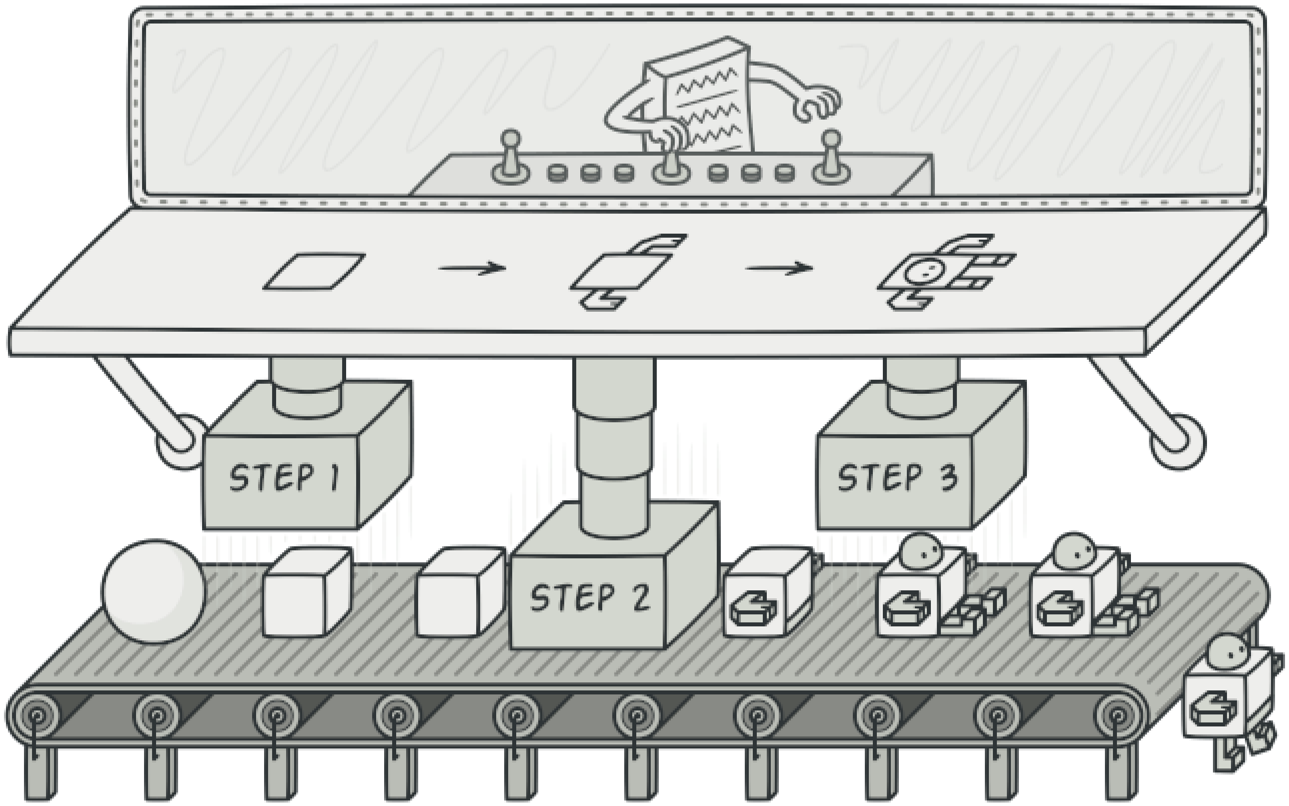

ConcreteFactory1
...
+ createProductA(): ProductA
+ createProductB(): ProductB
Concrete ProductA1
Concrete ProductB1
ProductA
ProductB
Concrete ProductA2
Concrete ProductB2
«interface»
AbstractFactory
+ createProductA(): ProductA
+ createProductB(): ProductB
Client
- factory: AbstractFactory
+ Client(f: AbstractFactory)
+ someOperation()
ProductA pa = factory.createProductA()
ConcreteFactory2
...
+ createProductA(): ProductA
+ createProductB(): ProductB
return new ConcreteProductA2()

Example


WinFactory
...
+ createButton(): Button
+ createCheckbox(): Checkbox
WinButton
WinCheckbox
Button
Checkbox
MacButton
MacCheckbox
«interface»
GUIFactory
+ createButton(): Button
+ createCheckbox(): Checkbox
Application
- factory: GUIFactory
- button: Button
...
+ Application(f: GUIFactory)
+ createUI()
+ paint()
MacFactory
...
+ createButton(): Button
+ createCheckbox(): Checkbox

Builder

Intent



Motivation: problem

Motivation: problem

Motivation: solution

Motivation: solution

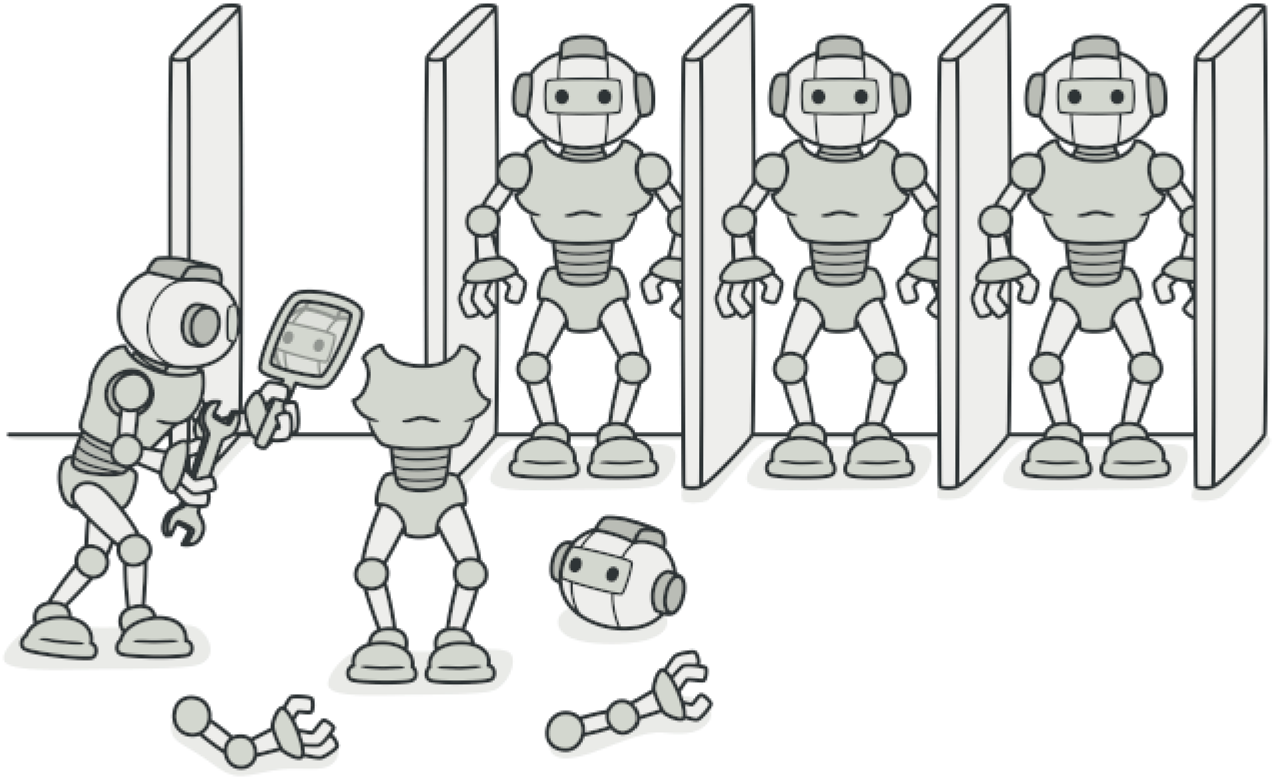
Motivation: solution

Structure

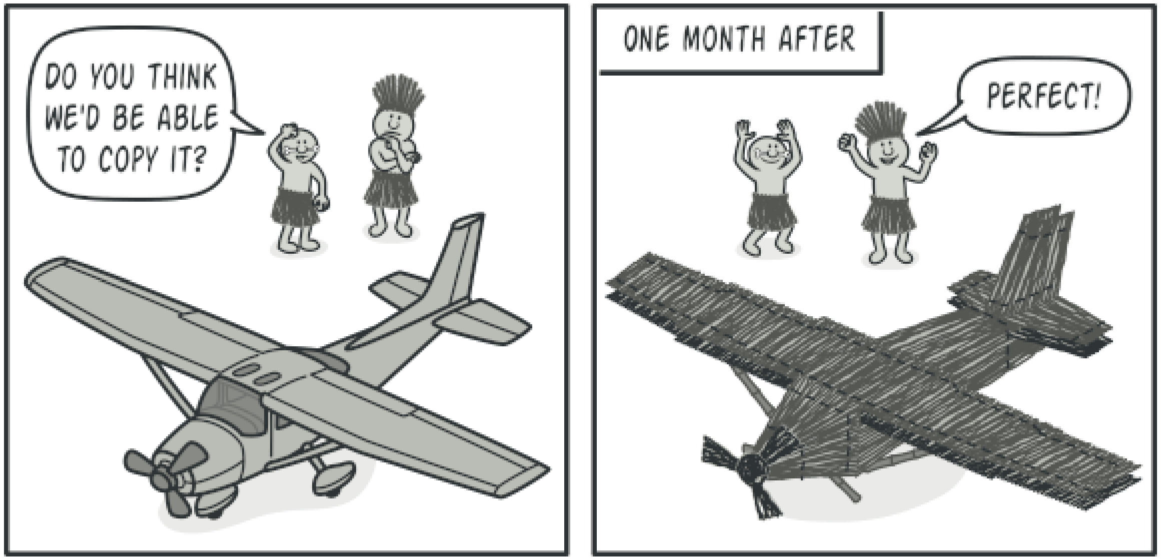
Example

Prototype

Intent

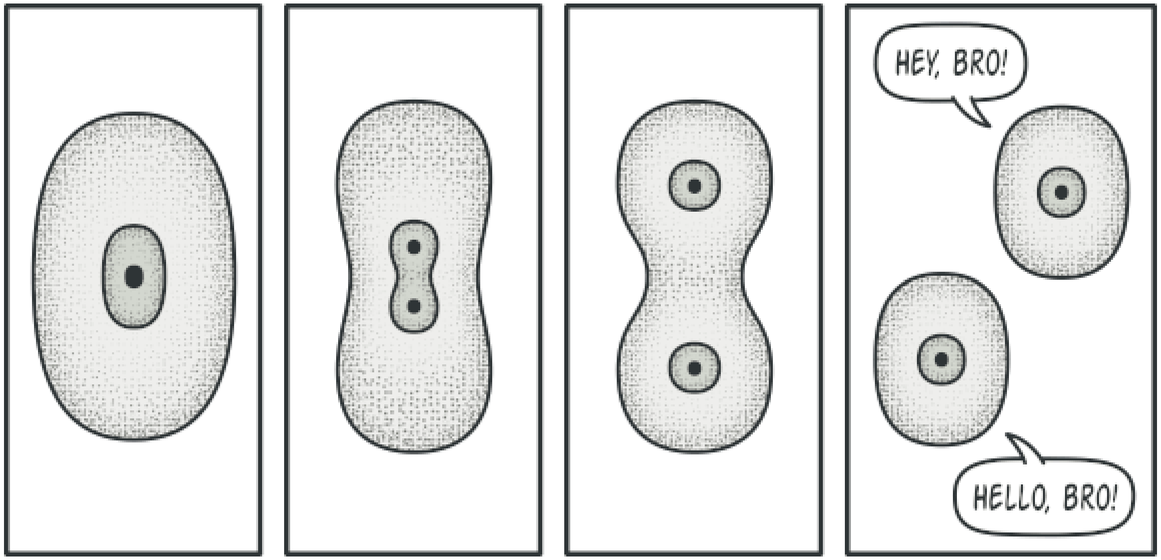


Motivation: problem

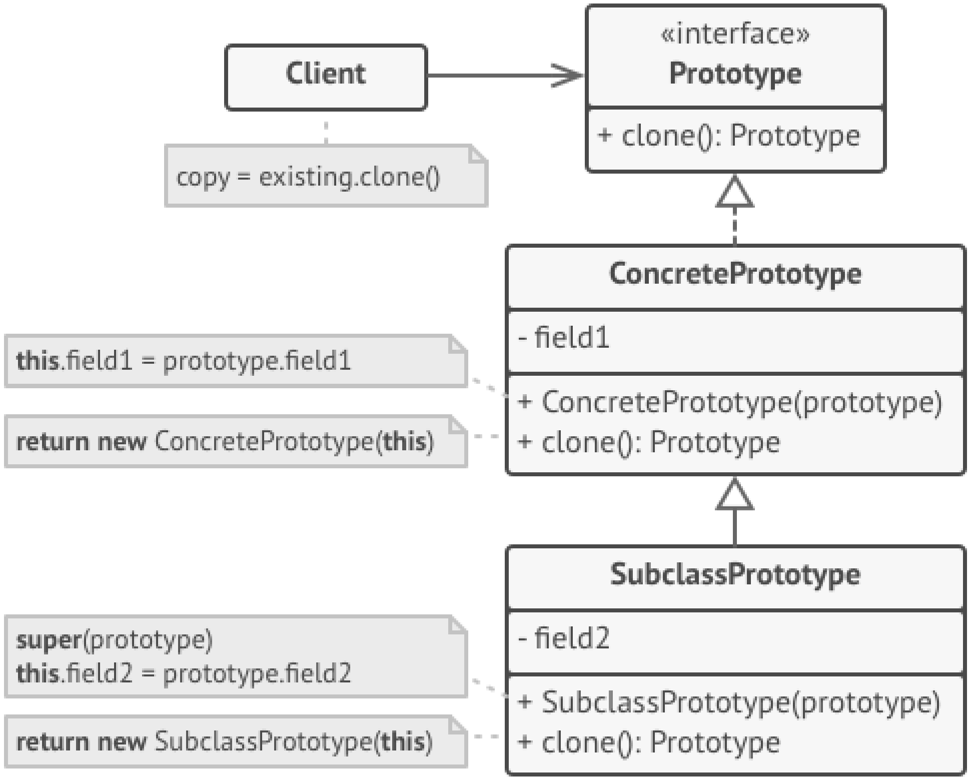


Motivation: solution

Motivation: solution

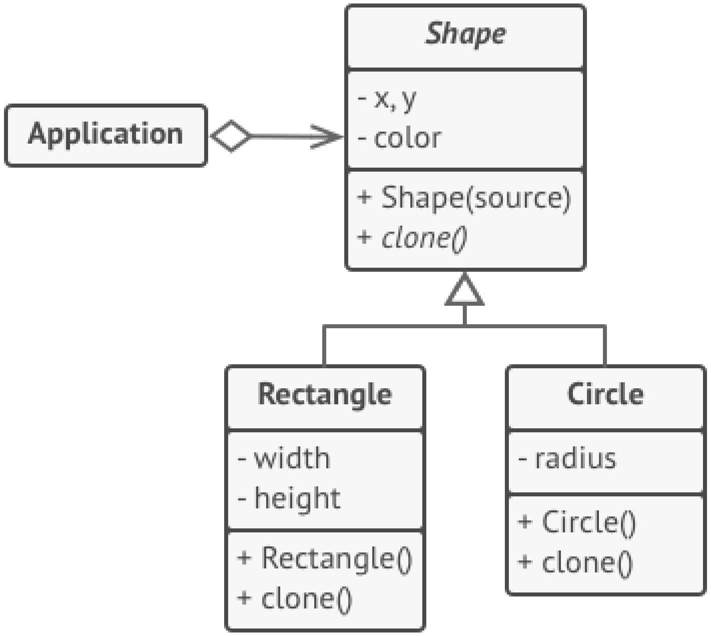


Structure


Client
copy = existing.clone()
«interface»
Prototype
+ clone(): Prototype
ConcretePrototype
- field1
+ ConcretePrototype(prototype)
+ clone(): Prototype
this.field1 = prototype.field1
return new ConcretePrototype(this)
SubclassPrototype
- field2
+ SubclassPrototype(prototype)
+ clone(): Prototype
super(prototype)
this.field2 = prototype.field2
return new SubclassPrototype(this)

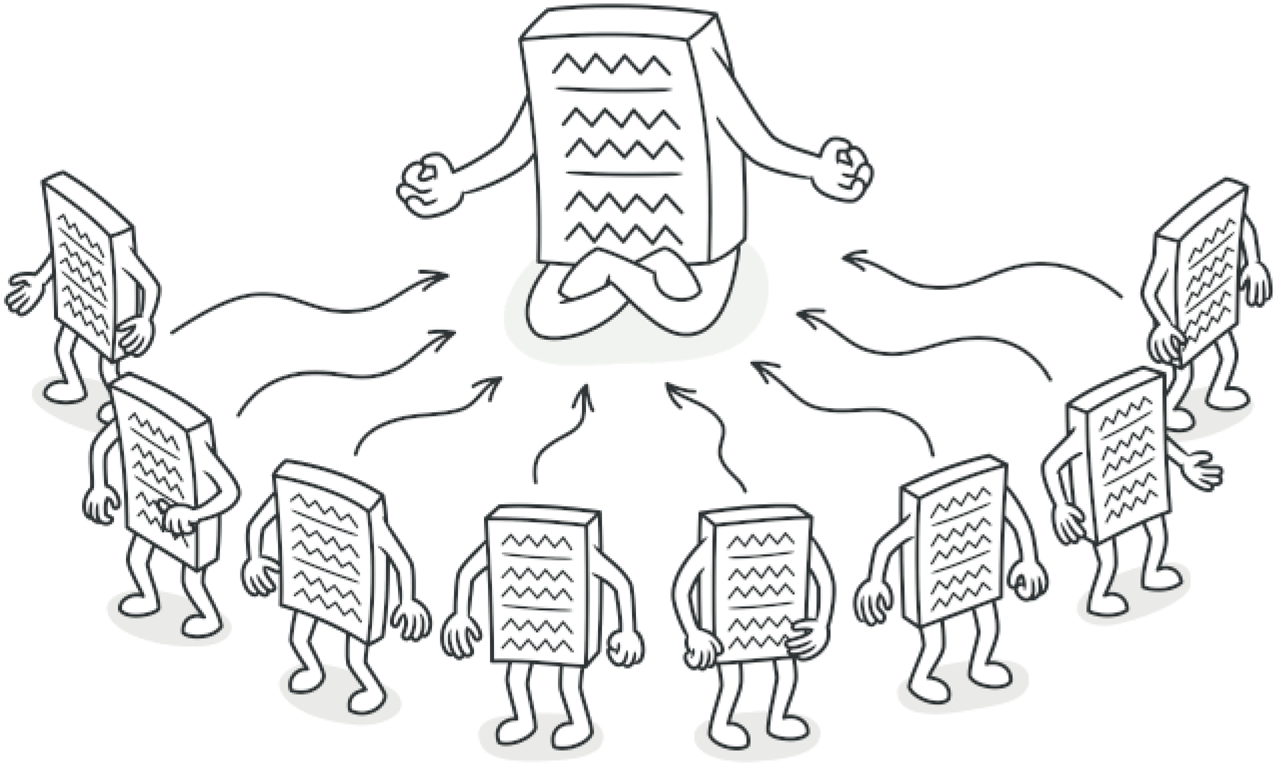
Structure

Example

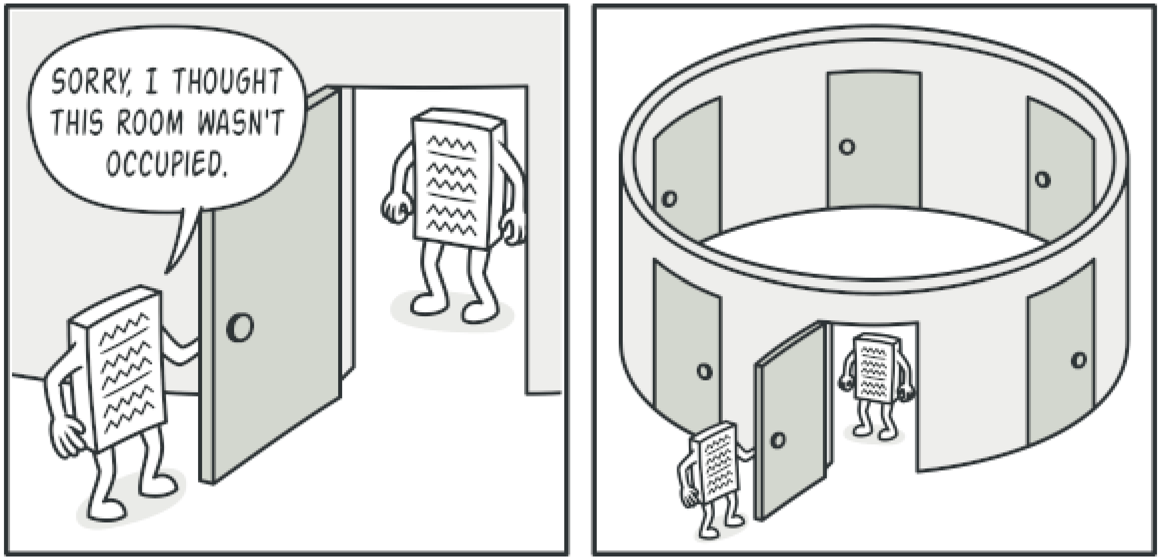

Application
Shape
- x, y
- color
+ Shape(source)
+ clone()
Rectangle
- width
- height
+ Rectangle()
+ clone()
Circle
- radius
+ Circle()
+ clone()

Singleton

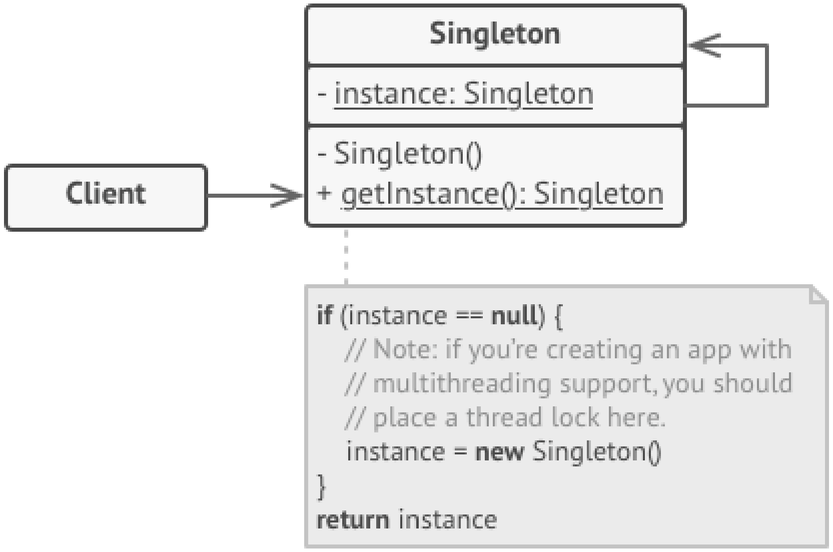
Intent



Motivation: problem



Structure



```
if (instance == null) {
   // Note: if you're creating an app with
   // multithreading support, you should
   // place a thread lock here.
   instance = new Singleton()
}
return instance
```

Structural patterns

Adapter

Intent

Motivation: problem

Motivation: solution

Motivation: solution

Structure

Structure

Example

Bridge

Intent

Motivation: problem

Motivation: solution

Motivation: solution

Motivation: solution

Structure

Example

Composite

Intent

Motivation: problem

Motivation: solution

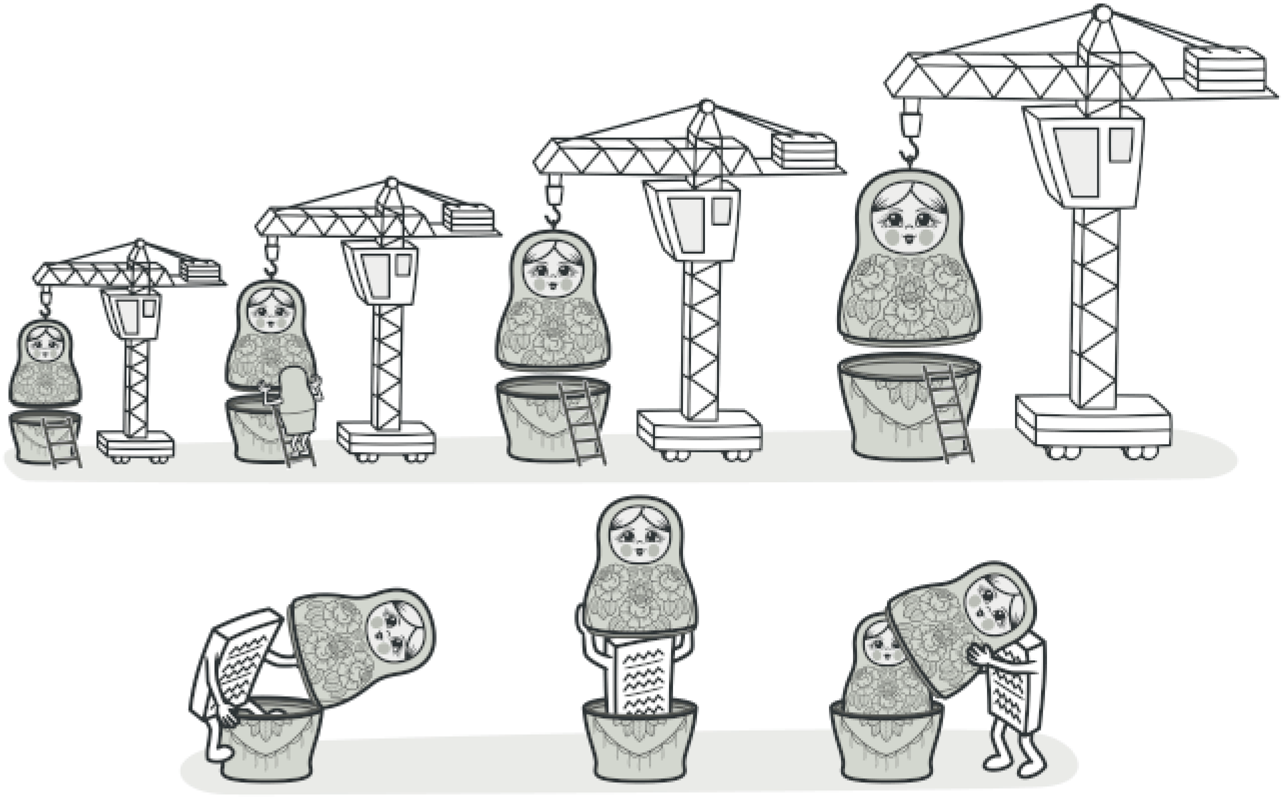
Motivation: solution

Structure

Example

Decorator

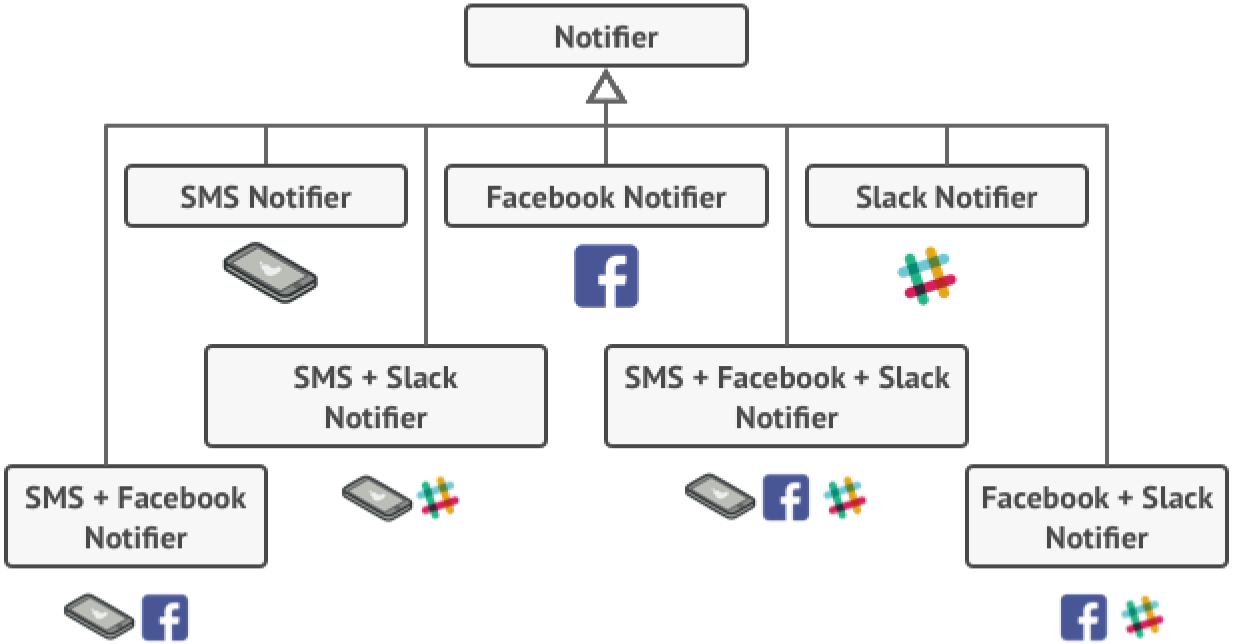
Intent



Motivation: problem

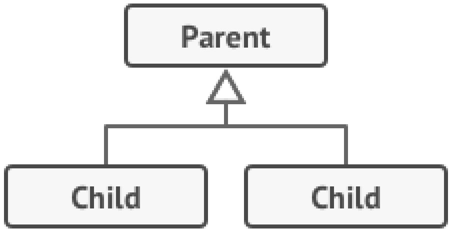
Motivation: problem

Motivation: problem

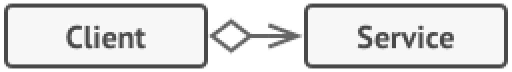


Motivation: solution

Inheritance

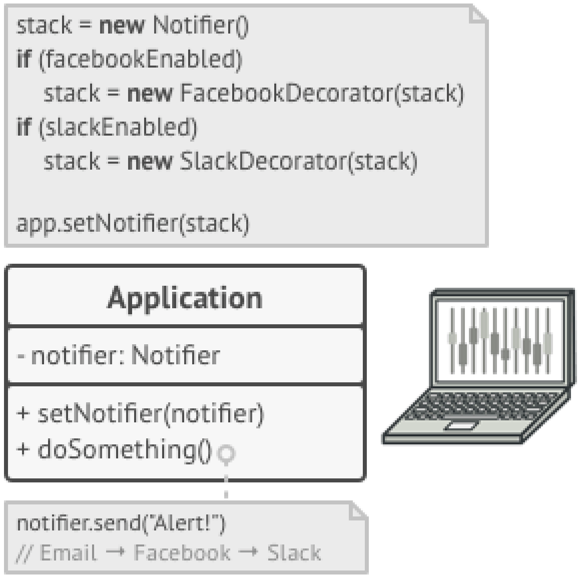


Aggregation



Motivation: solution

Motivation: solution



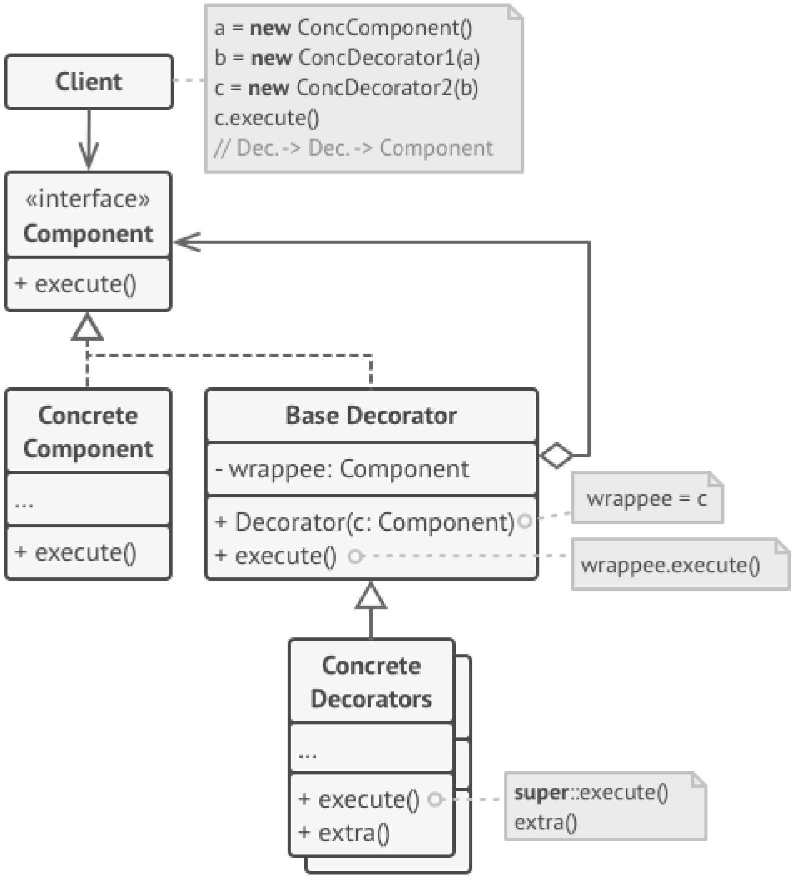
```
stack = new Notifier()
if (facebookEnabled)
    stack = new FacebookDecorator(stack)
if (slackEnabled)
    stack = new SlackDecorator(stack)

app.setNotifier(stack)
```

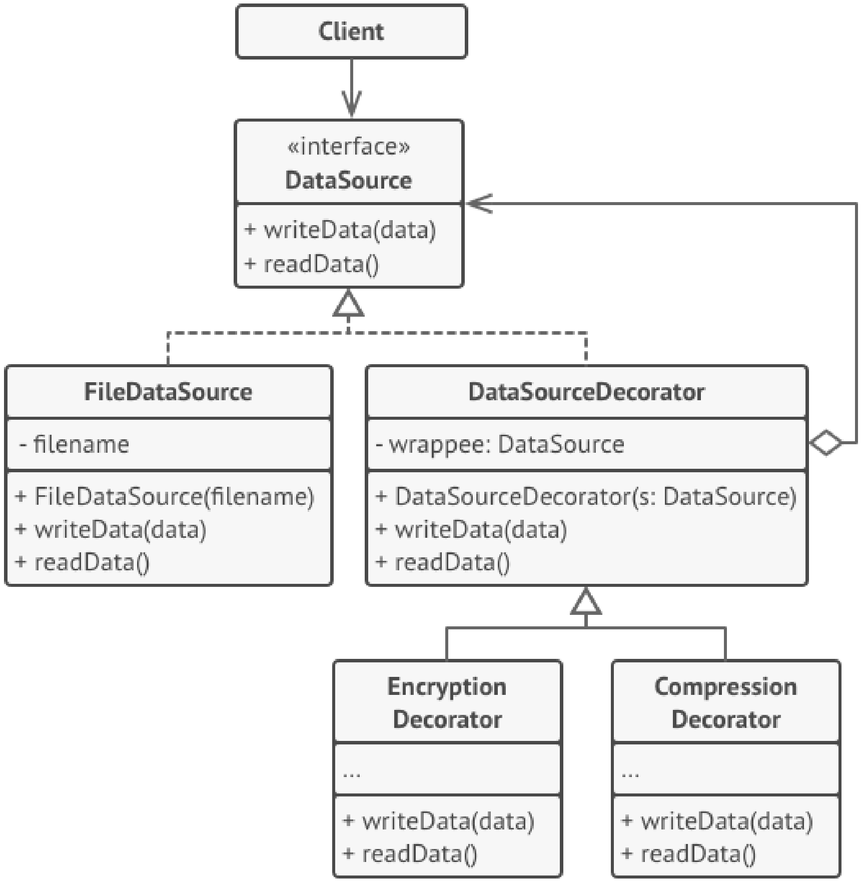
```
notifier.send("Alert!")
// Email → Facebook → Slack
```

Motivation: solution

Structure

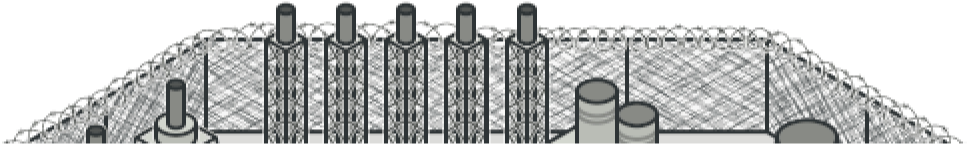

Client
a = new ConcComponent()
b = new ConcDecorator1(a)
c = new ConcDecorator2(b)
c.execute()
// Dec. -> Dec. -> Component
«interface»
Component
+ execute()
Concrete
Component
...
+ execute()
Base Decorator
- wrappee: Component
+ Decorator(c: Component)
+ execute()
wrappee = c
wrappee.execute()
Concrete
Decorators
...
+ execute()
+ extra()
super::execute()
extra()

Example


Client
«interface»
DataSource
+ writeData(data)
+ readData()
FileDataSource
- filename
+ FileDataSource(filename)
+ writeData(data)
+ readData()
DataSourceDecorator
- wrappee: DataSource
+ DataSourceDecorator(s: DataSource)
+ writeData(data)
+ readData()
Encryption Decorator
...
+ writeData(data)
+ readData()
Compression Decorator
...
+ writeData(data)
+ readData()

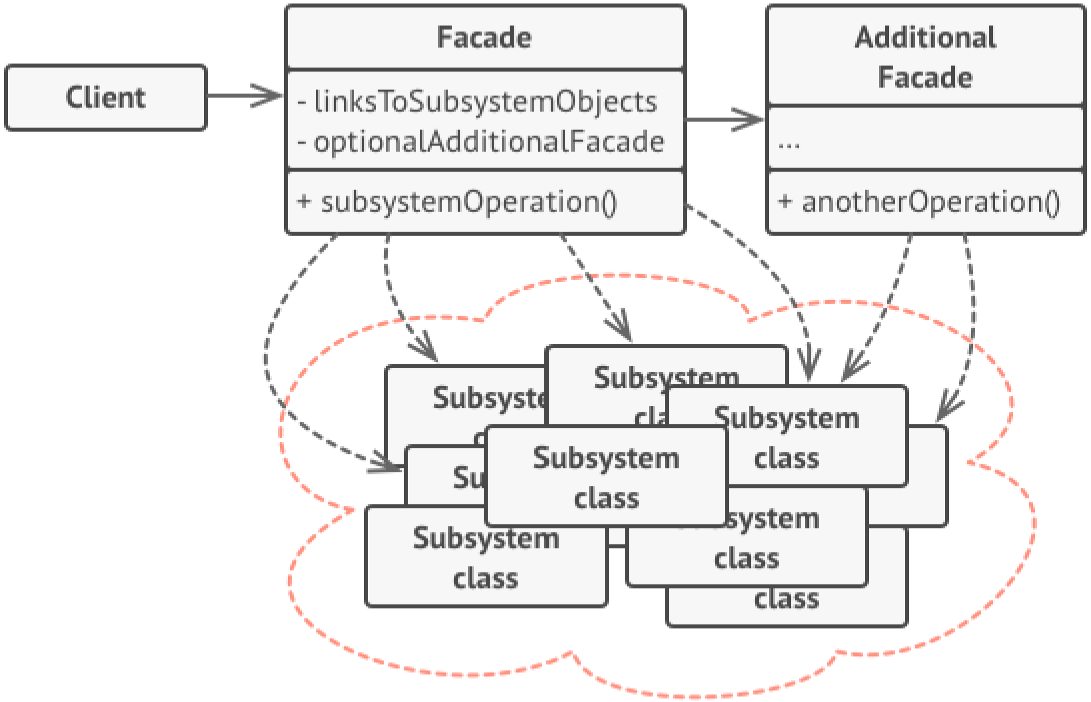
Facade

Intent



Motivation: solution

Structure

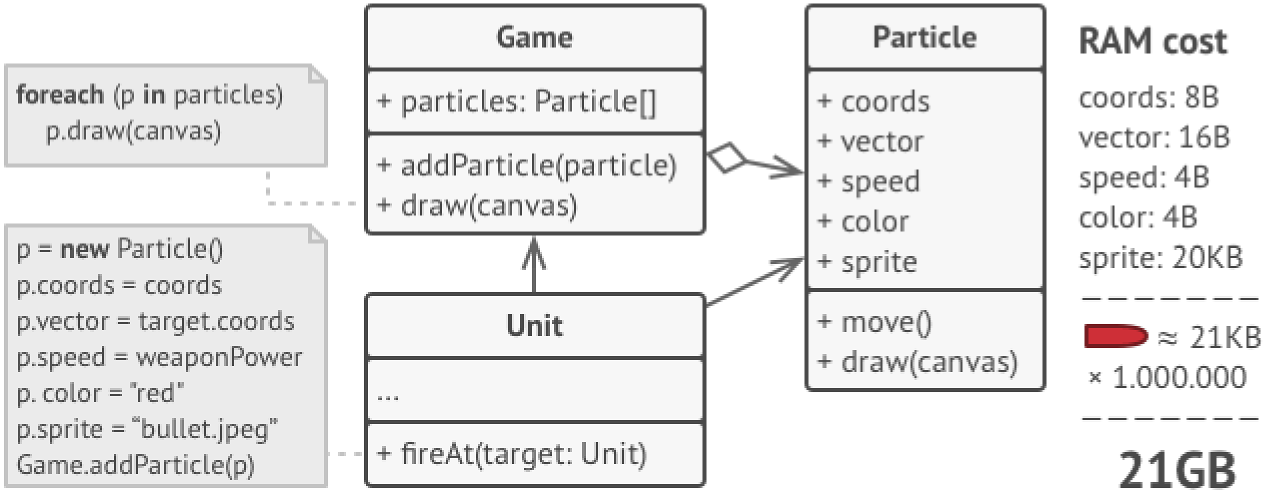

Client
Facade
- linksToSubsystemObjects
- optionalAdditionalFacade
+ subsystemOperation()
Additional Facade
...
+ anotherOperation()
Subsystem class
Subsystem class
Subsystem class
Subsystem class
Subsystem class
Subsystem class
Subsystem class

Example

Flyweight

Intent

Motivation: problem



```
foreach (p in particles)
    p.draw(canvas)
```

```
p = new Particle()
p.coords = coords
p.vector = target.coords
p.speed = weaponPower
p. color = "red"
p.sprite = "bullet.jpeg"
Game.addParticle(p)
```

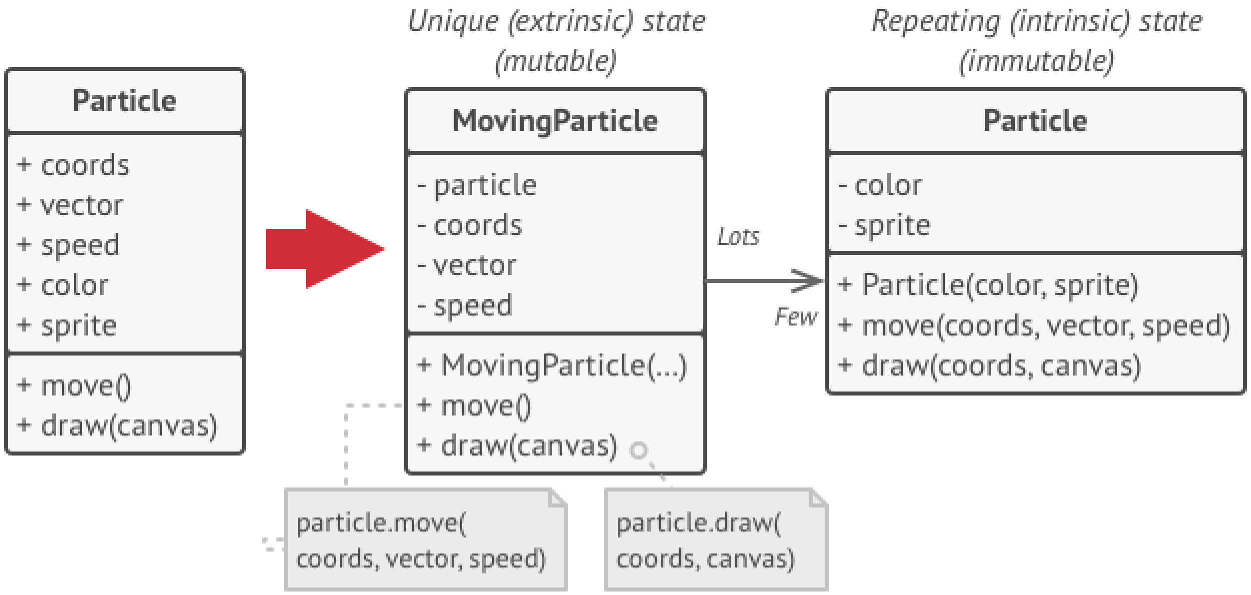
RAM cost

coords: 8B
vector: 16B
speed: 4B
color: 4B
sprite: 20KB

≈ 21KB
× 1.000.000

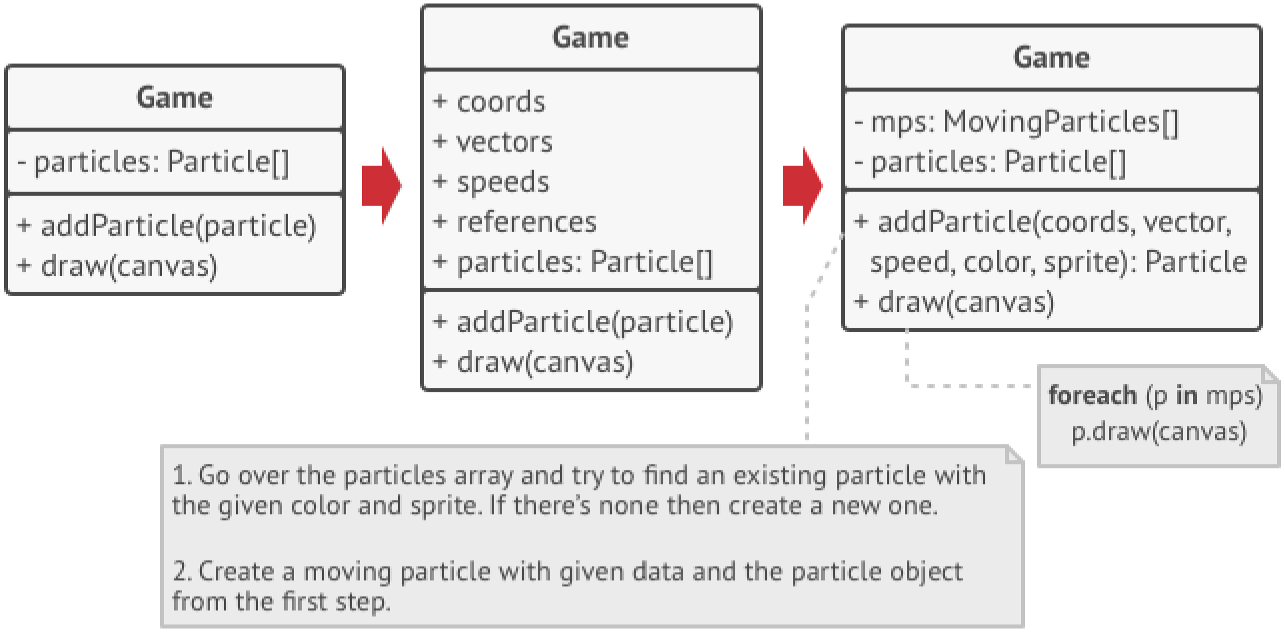
21GB

Motivation: solution



Motivation: solution

Motivation: solution



1. Go over the particles array and try to find an existing particle with the given color and sprite. If there's none then create a new one.

2. Create a moving particle with given data and the particle object from the first step.

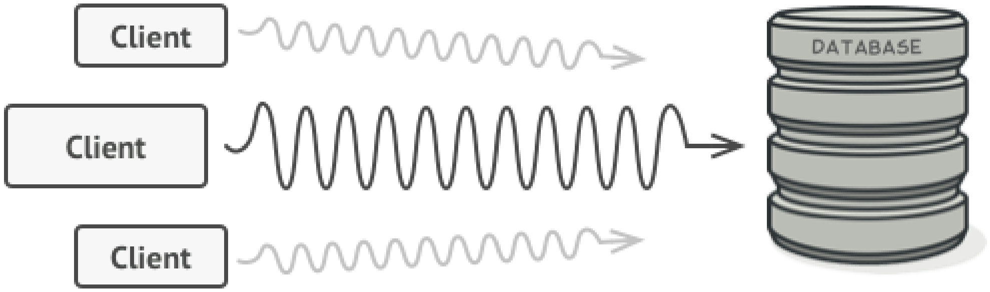
Structure

Example

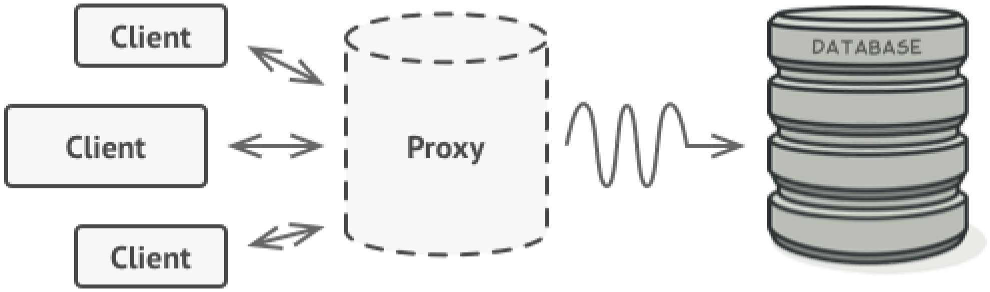
Proxy

Intent

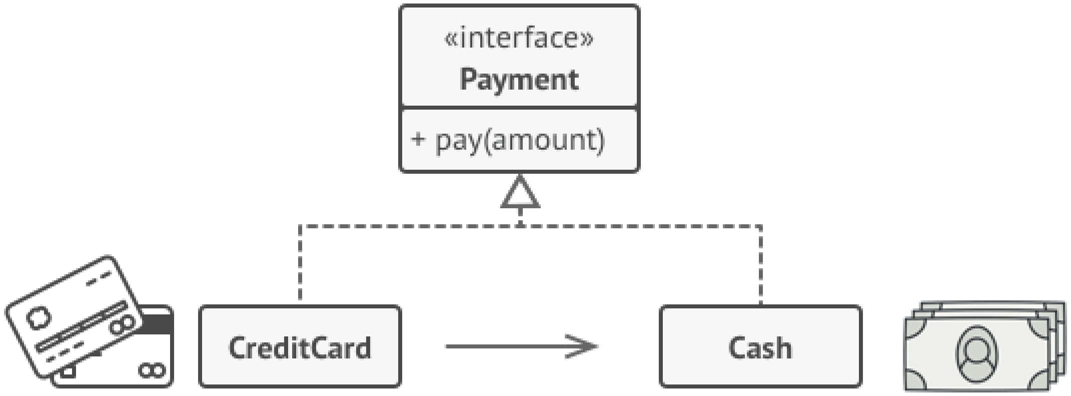
Motivation: problem



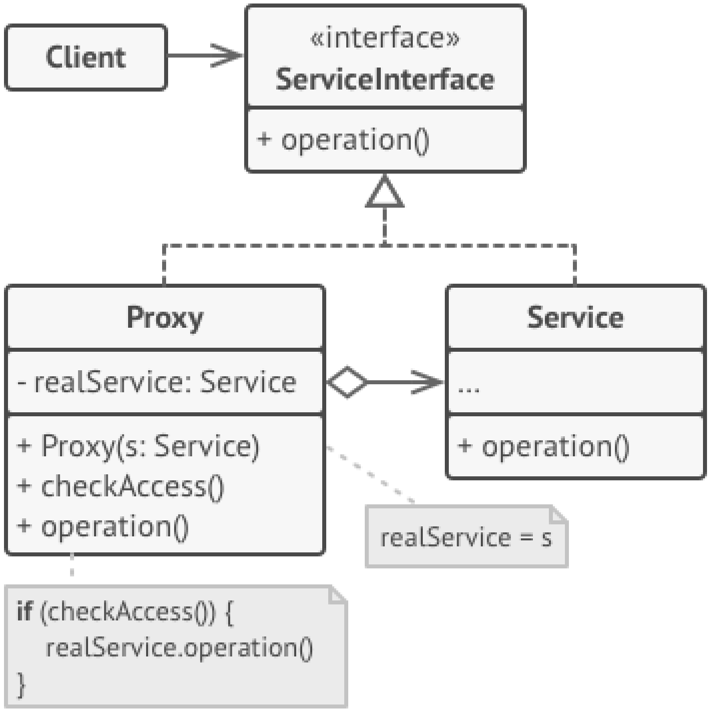
Motivation: solution



Motivation: solution



Structure


Client
«interface»
ServiceInterface
+ operation()
Proxy
- realService: Service
+ Proxy(s: Service)
+ checkAccess()
+ operation()
Service
...
+ operation()
realService = s
if (checkAccess()) {
realService.operation()
}

Example

Behavioral patterns

Chain of Responsibility

Intent

Motivation: problem

Motivation: problem

Motivation: solution

Motivation: solution

Motivation: solution

Structure

Example

Example

Command

Intent

Motivation: problem

Motivation: problem

Motivation: problem

Motivation: solution

Motivation: solution

Motivation: solution

Motivation: solution

Structure

Example

Iterator

Intent

Motivation: problem

Motivation: problem

Motivation: solution

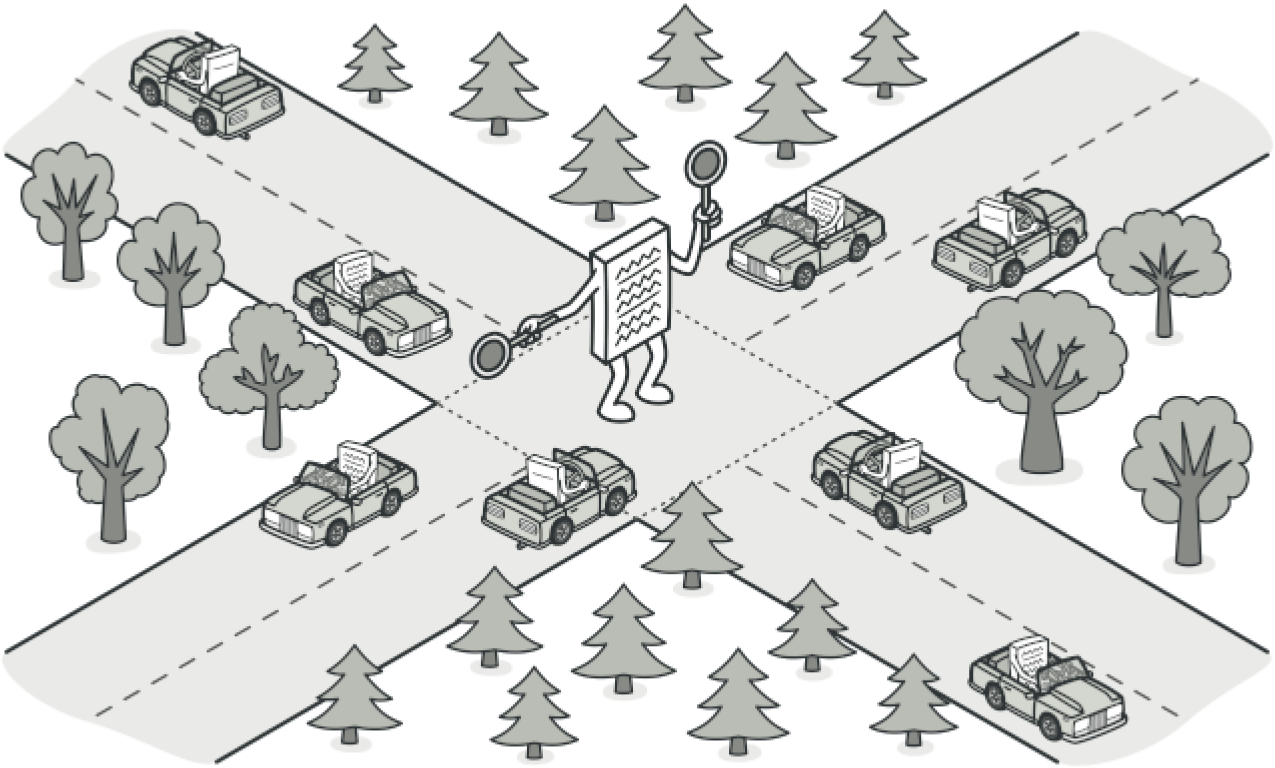
Motivation: solution

Structure

Example

Mediator

Intent



Motivation: problem

Motivation: problem

Motivation: solution

Motivation: solution

Structure

Example

Momento

Intent

Motivation: problem

Motivation: problem

Motivation: solution

Structure

Structure

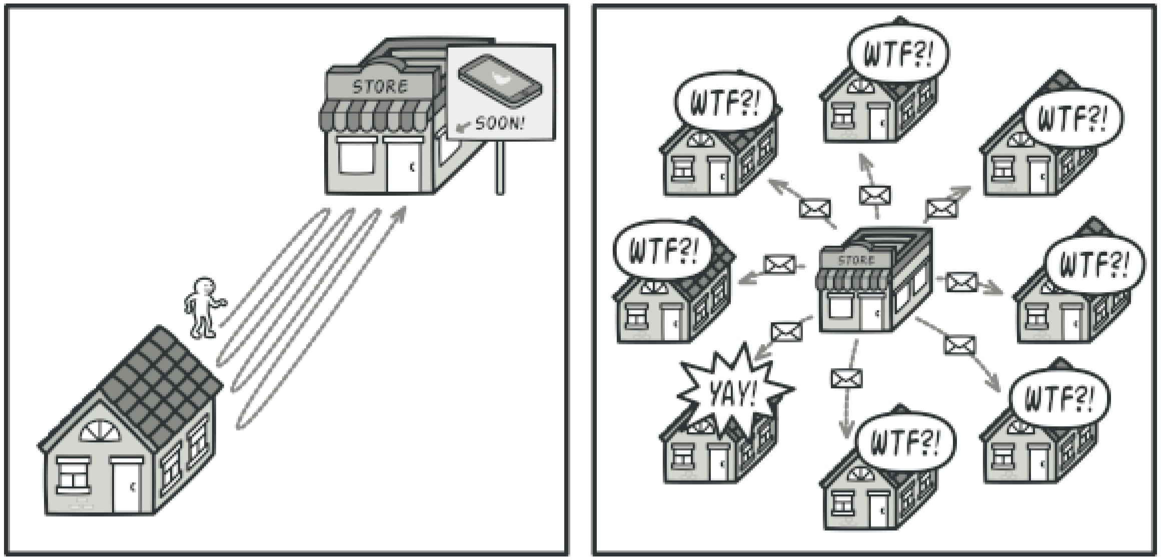
Structure

Example

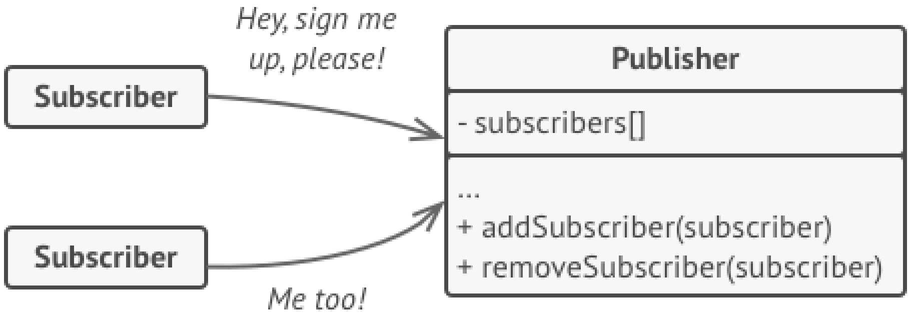
Observer

Intent

Motivation: problem



Motivation: solution



Motivation: solution

Motivation: solution

Structure

Example

State

Intent

Motivation: problem

Motivation: problem

Motivation: solution

Structure

Example

Strategy

Intent

Motivation: problem

Motivation: solution

Motivation: solution

Structure

Template Method

Intent

Motivation: problem

Motivation: solution

Motivation: solution

Structure

Example

Visitor

Intent

Motivation: problem

Motivation: problem

Motivation: solution

Structure

Example