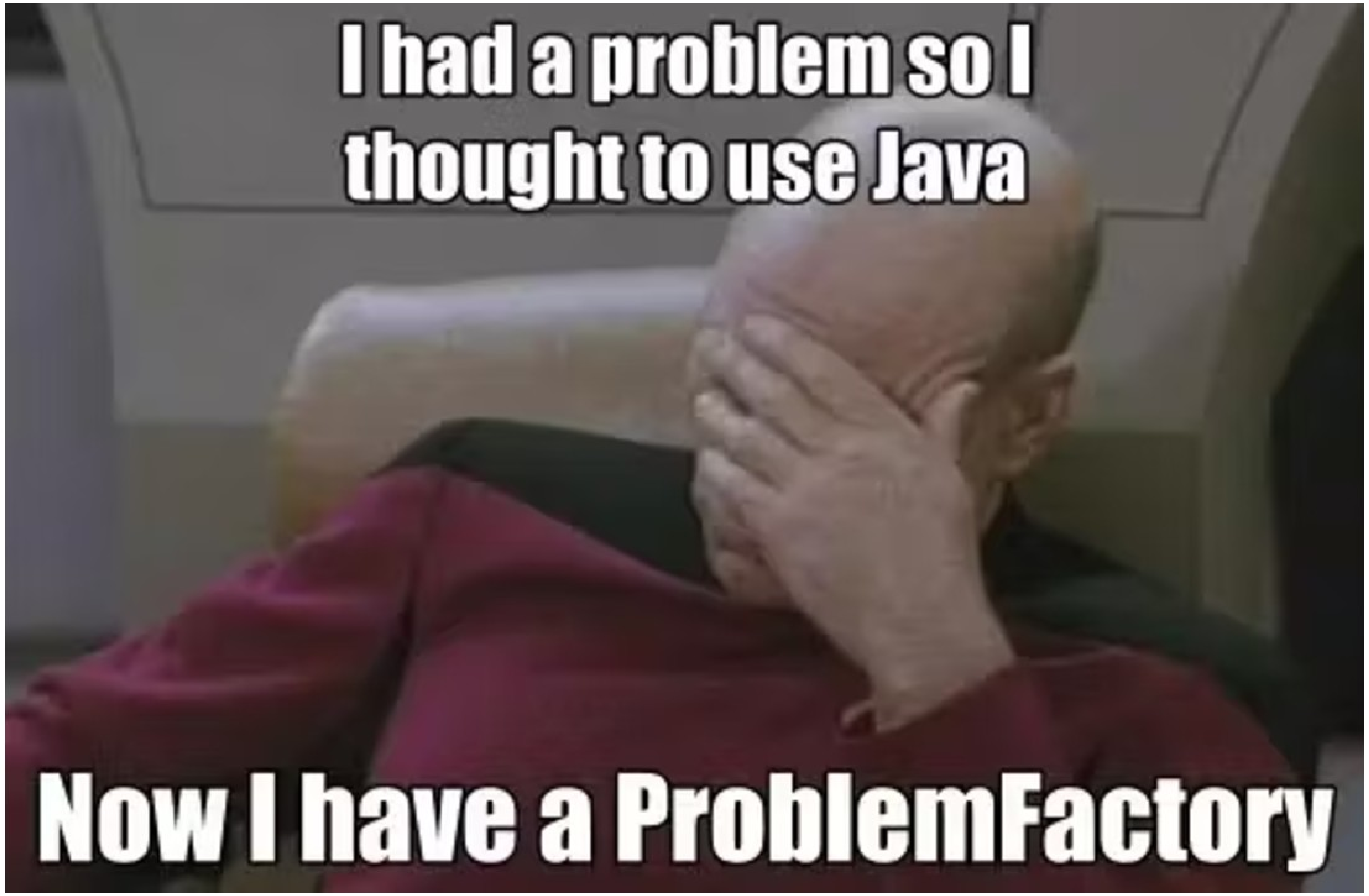
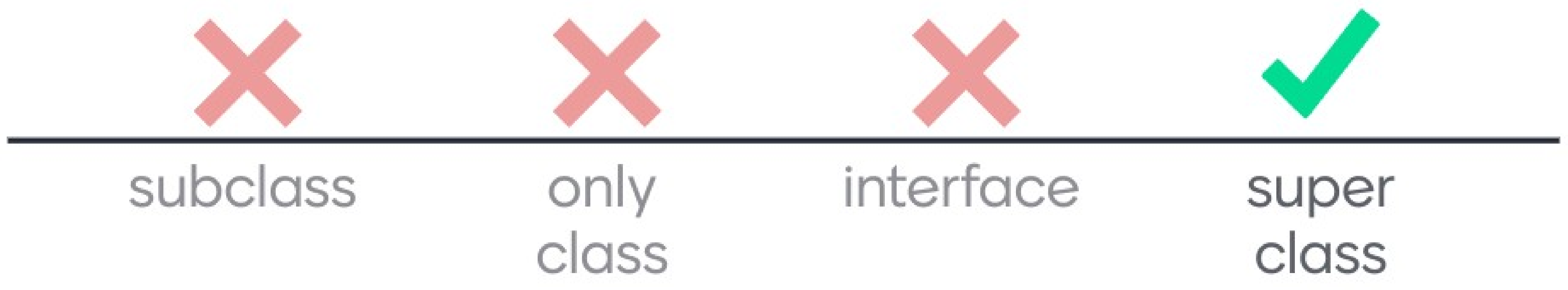
Mentimeter


I had a problem so I thought to use Java
Now I have a ProblemFactory

The factory pattern relies heavily on what principle?

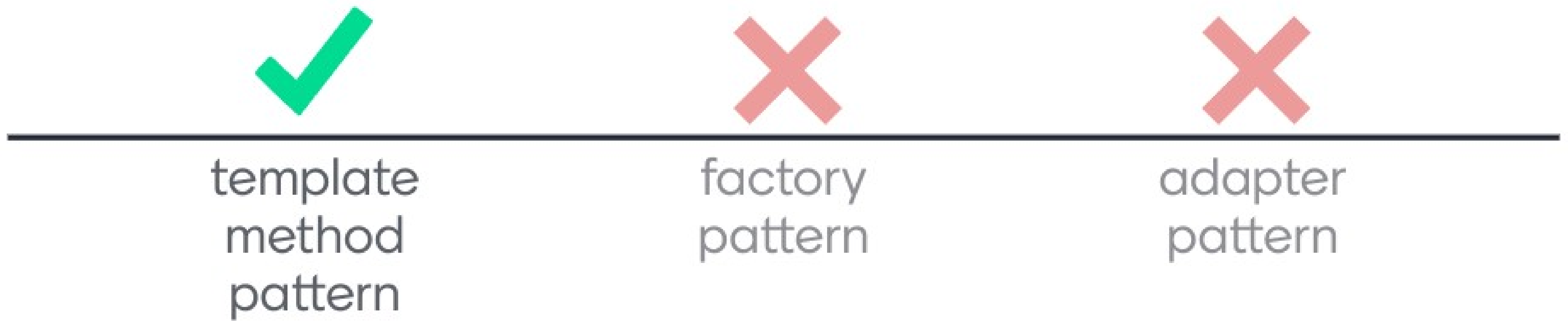
✗	✗	✓
single responsibility principle	interface segregation principle	liskov substitution principle

The template method for the template method pattern goes in the

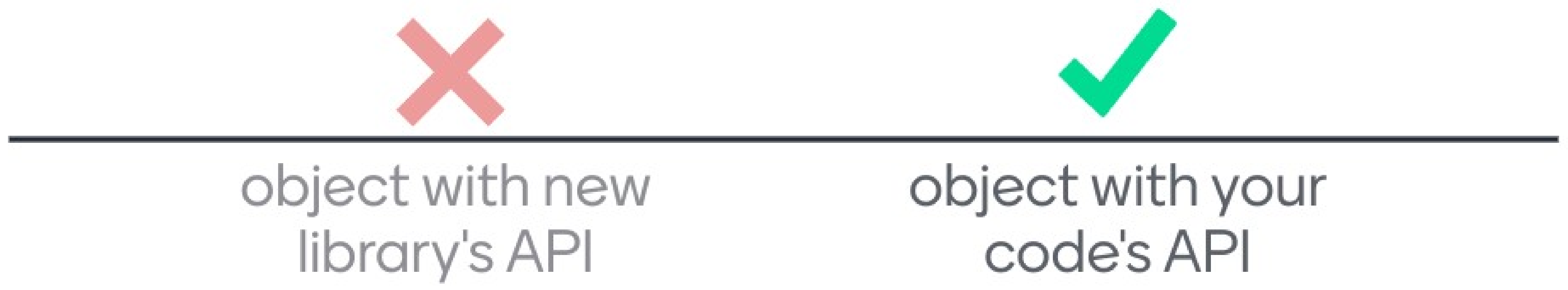


Mentimeter

"hooks" are similar to

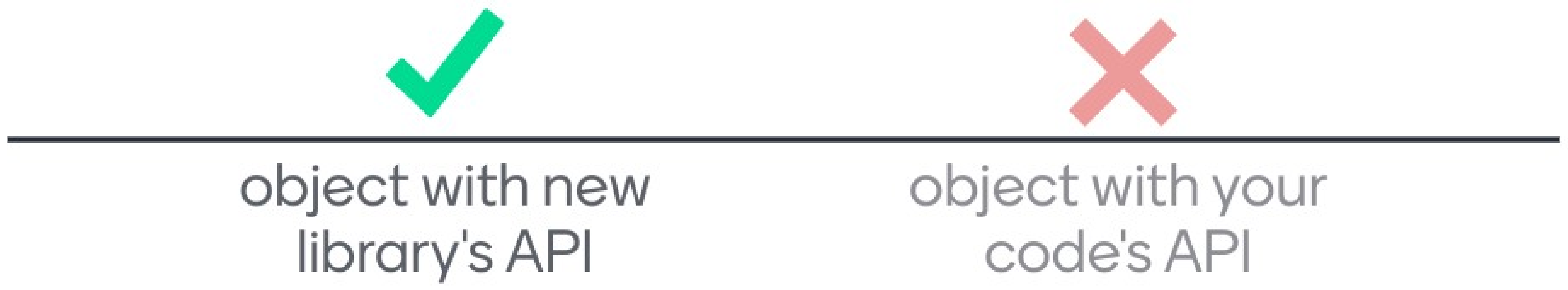


In the adapter pattern, the client (caller) has a reference to...



Mentimeter

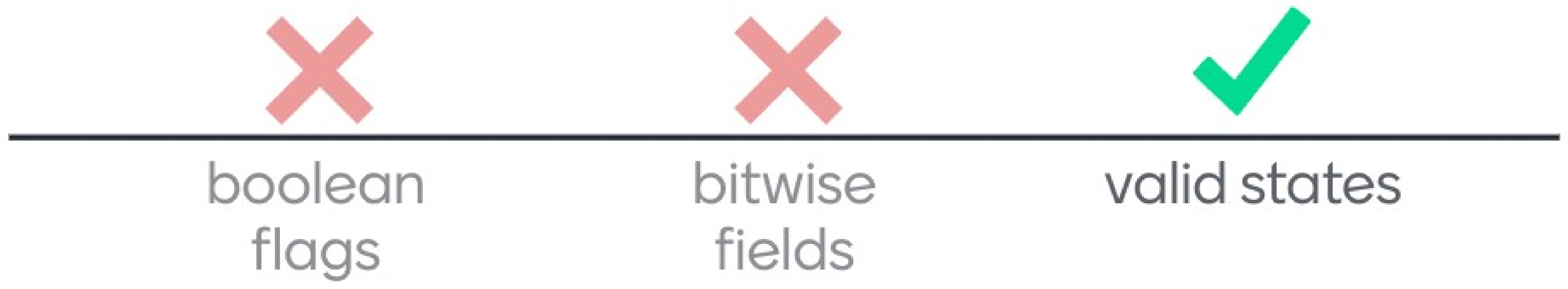
In the adapter pattern, adapter object has a reference to...



In the adapter pattern

✗	✓	✗
creat a new method	create a new class	create a new interface

In the state pattern we explicitly define

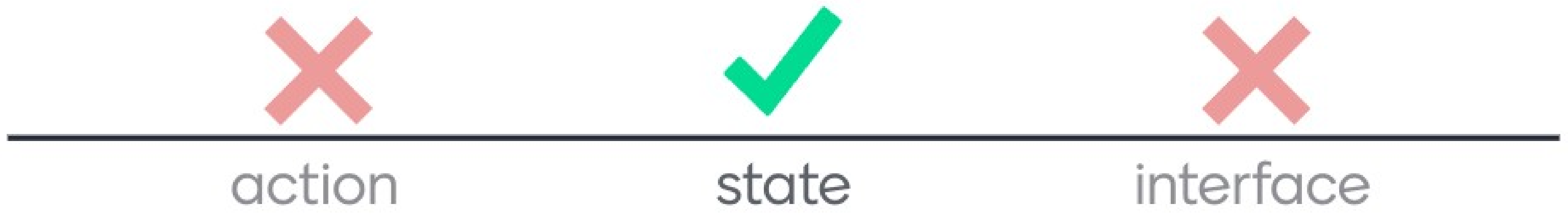


In the state pattern, we explicitly define

✓	✗	✗
each state to state transition	invalid states	valid combinations of states

Mentimeter

In the state pattern, the result of every action in every state is grouped by



The state pattern helps us avoid...

✓ Invalid states

✓ Forgotten combinations

✗ Null pointer exceptions

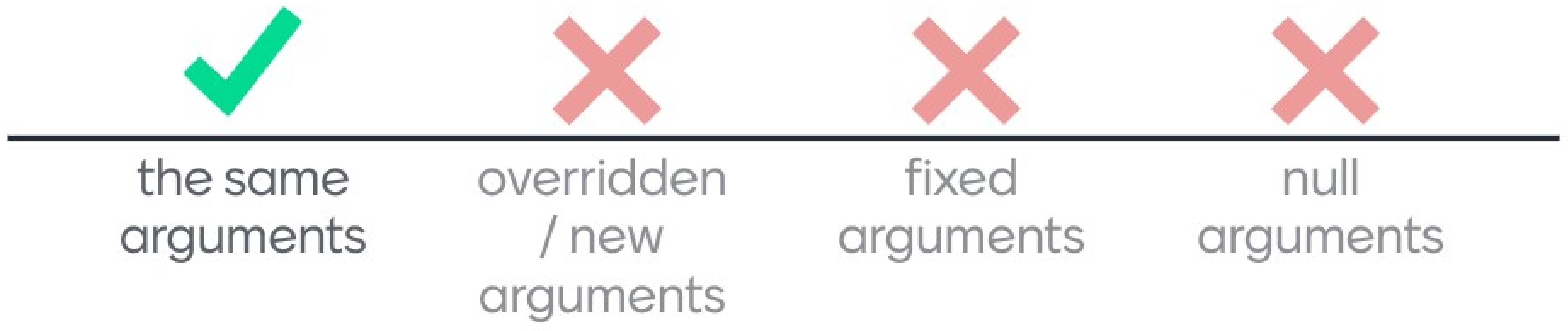
Mentimeter

In the proxy pattern most methods

✓	✗	✗
call the same method on the attribute	are empty / do nothing / raise an error	are abstract

Mentimeter

In the proxy pattern when we call the same method we usually call it with...



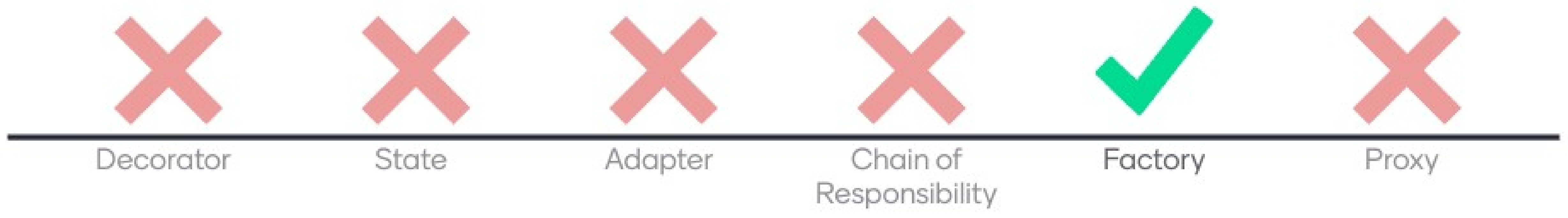
Mentimeter

In the proxy pattern when we call the same method we usually return...

✓	✗	✗	✗
the same return value	overridden / new return value	fixed return value	null return value

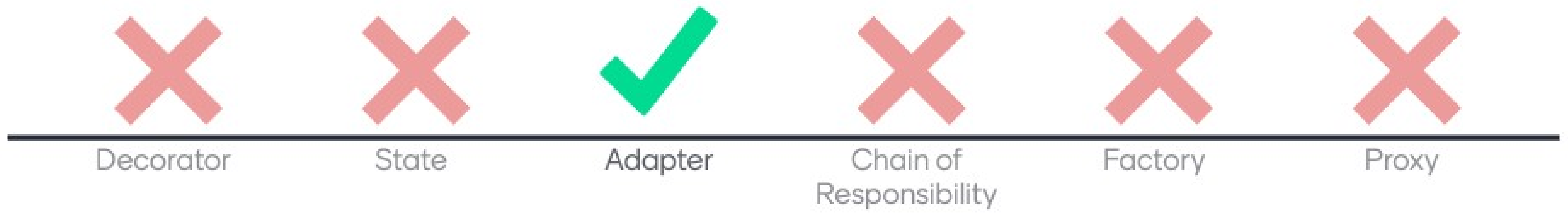
Mentimeter

Let a specialized class decide which subclass to create



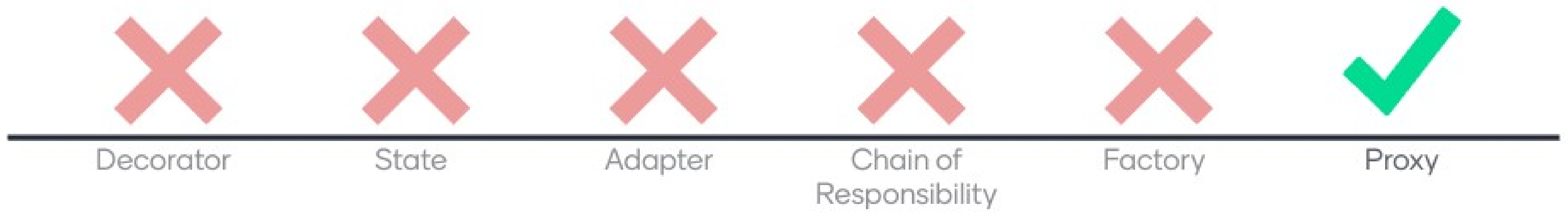
Mentimeter

I need to get two incompatible classes to work together...



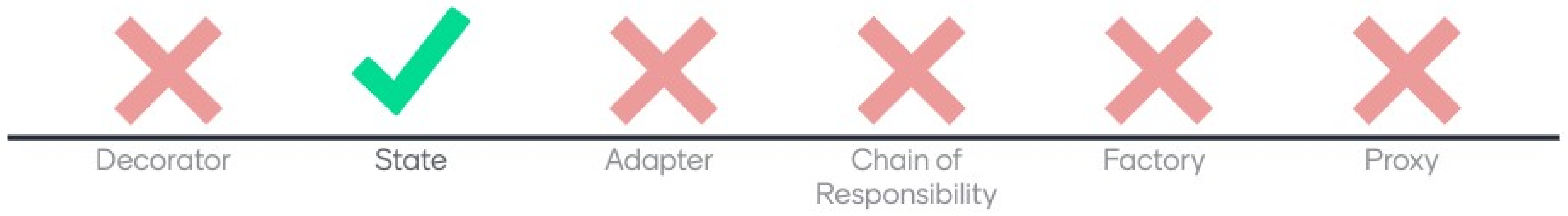
Mentimeter

I need create a small/fast object to cache results, only call the original heavy/big/slow object when needed...



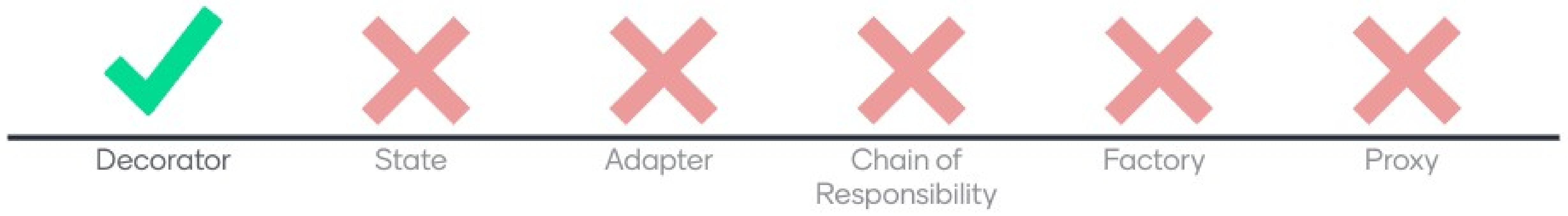
Mentimeter

I need to simplify and organize methods with long conditionals on combinations of booleans...



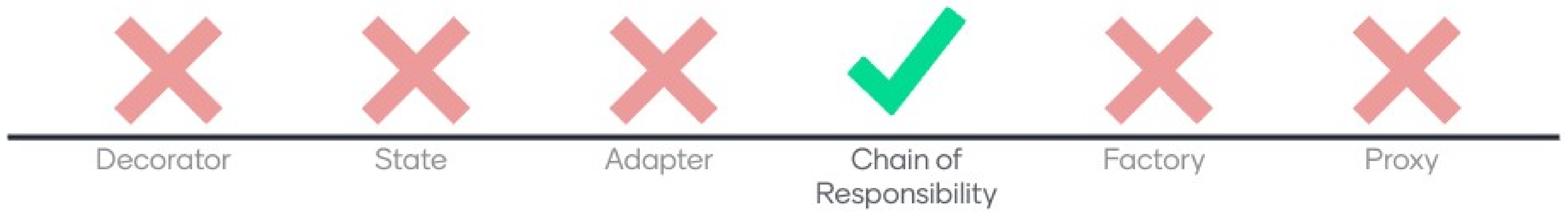
Mentimeter

I need to use a single interface but add aditional responsibilities to an object at run time...

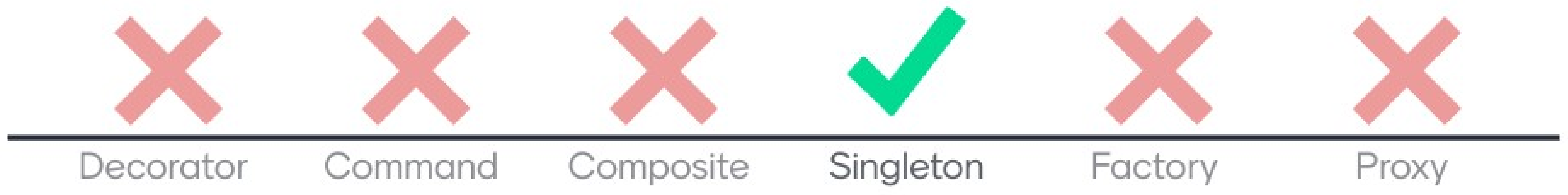


Mentimeter

Find the correct object for a task by passing along the same arguments from object to object...

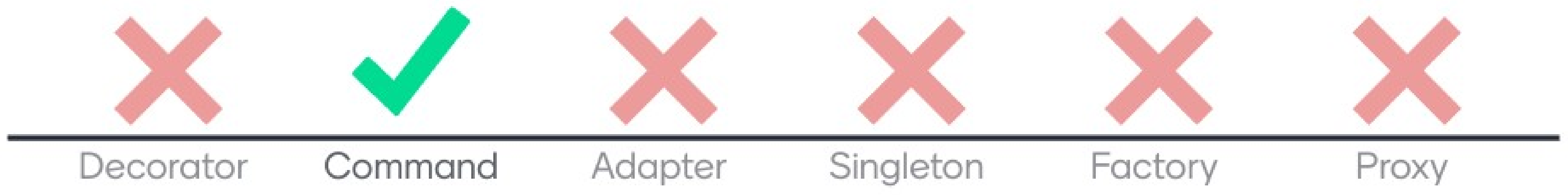


I need something like a gobal variablebut without making a global variable...



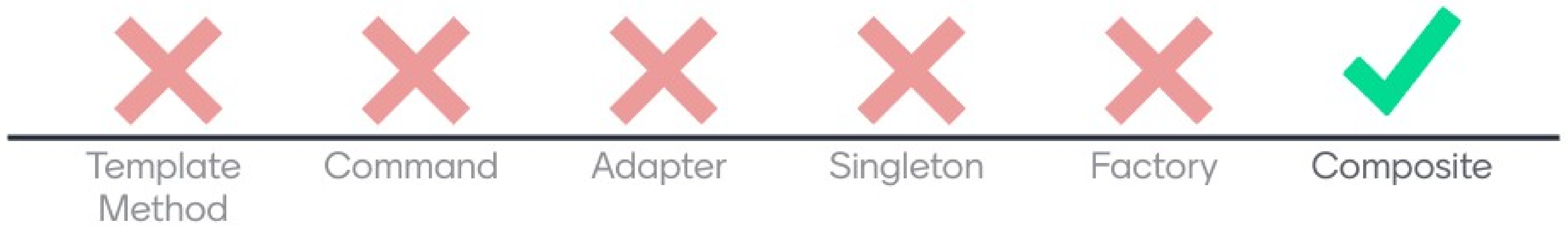
Mentimeter

I need something so I can replay a sequence of steps...



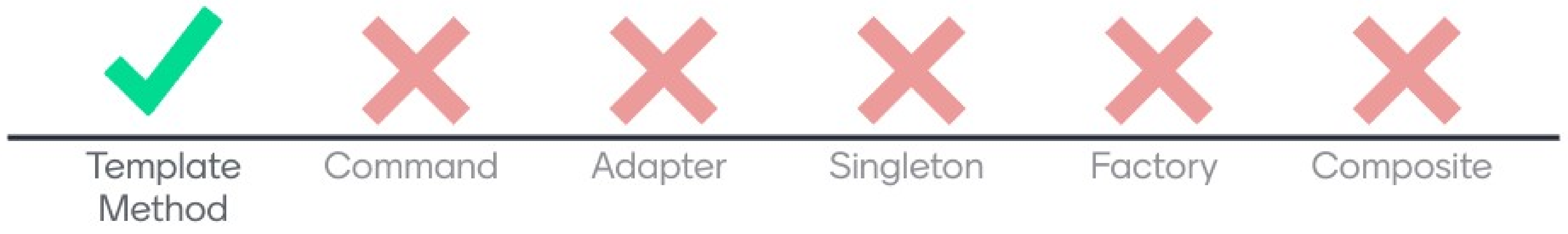
Mentimeter

A tree of different things but all having the same interface...



Mentimeter

Customized behavior by overriding steps in a subclass...



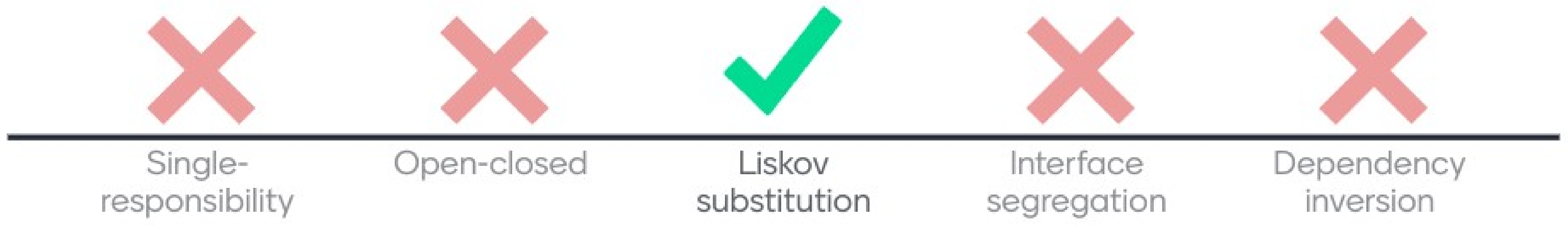
Mentimeter

What's the first OO design principle we learned?

✓	✗	✓	✗	✗
Single-responsibility	Open-closed	Liskov substitution	Interface segregation	Dependency inversion

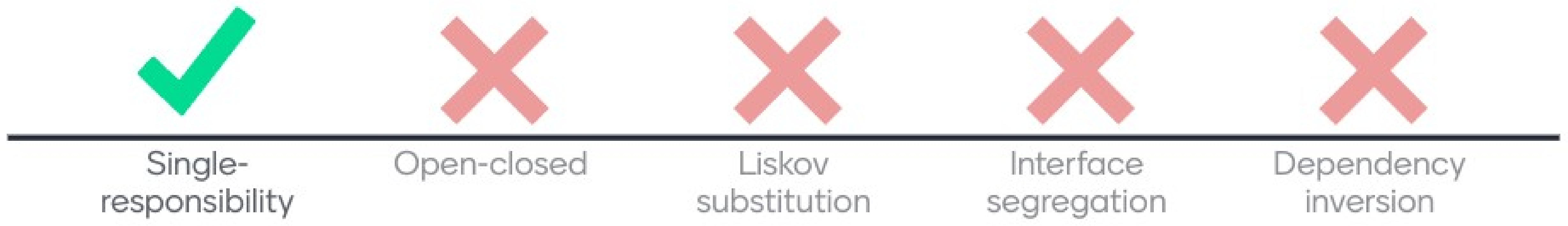
Mentimeter

What principle says we should be able to use a subclass anywhere we can use the superclass without noticing?



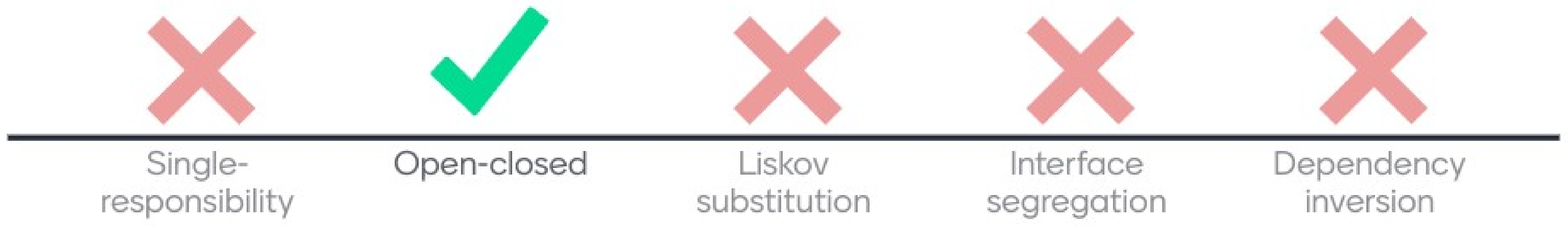
Mentimeter

What principle says that a class should only do one thing?



Mentimeter

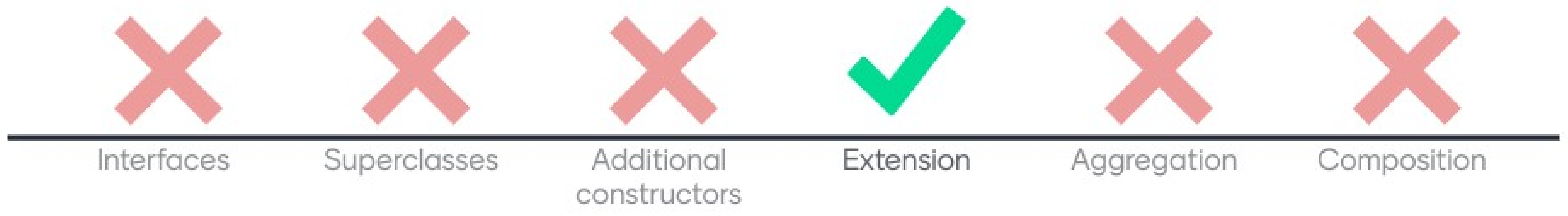
What principle says that a class shouldn't change once it's well tested and has good clean code?



✗	✓	✗	✗	✗
Single-responsibility	Open-closed	Liskov substitution	Interface segregation	Dependency inversion

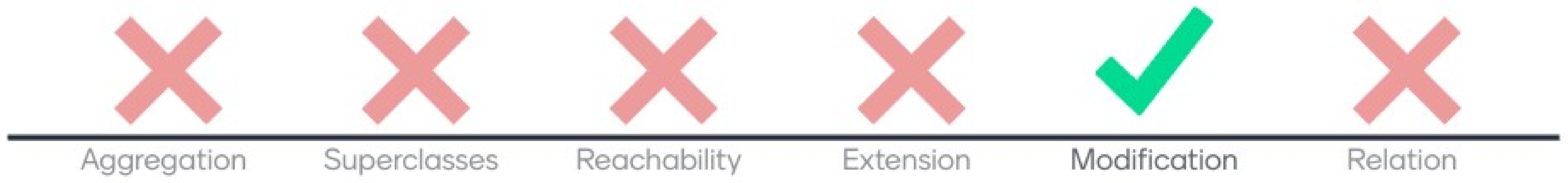
Mentimeter

In open-closed what should the class be open to?



Mentimeter

In open-closed what should the class be closed to?

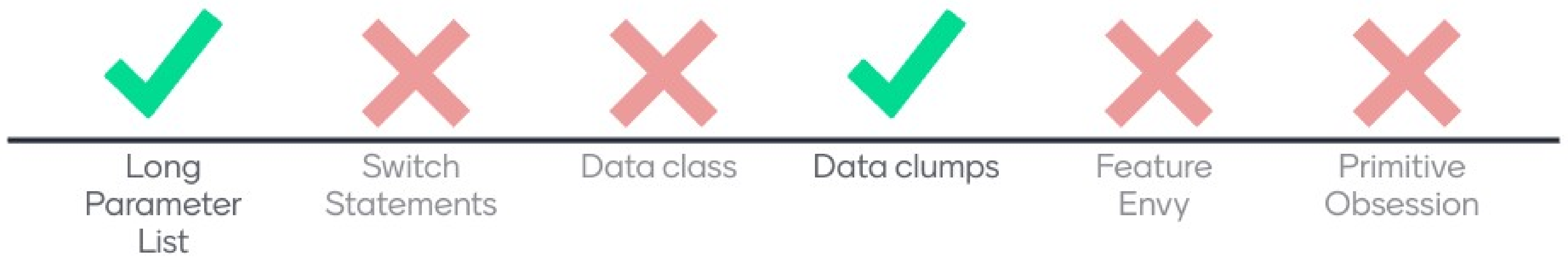


Mentimeter

What is the principle of least knowledge?

- ✓ Limit the classes whose methods we call
- ✗ Limit the number of public classes
- ✗ Limit the number of instance attributes/ methods
- ✗ Limit the number of static attributes/ methods

Can be fixed by making a class to store multiple, related parameters

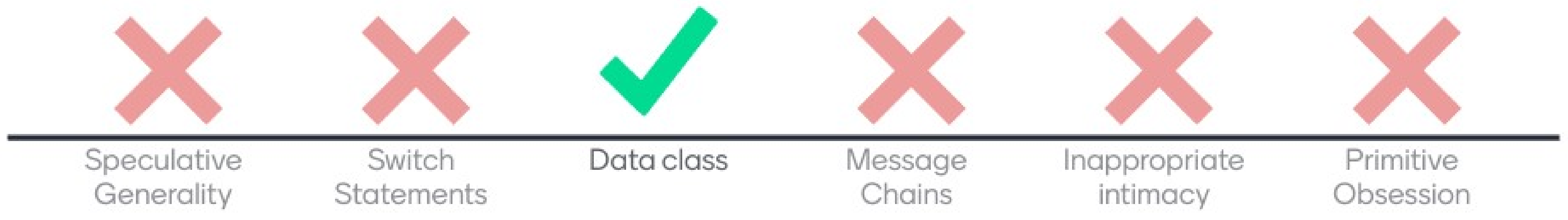


A method seems more interested in the details of another class than the one it is actually in

✗	✗	✗	✗	✓	✗
Long Parameter List	Switch Statements	Data class	Data clumps	Feature Envy	Primitive Obsession

Mentimeter

Classes that are just attributes with no real methods



fix it with a state pattern

✗	✓	✗	✗	✗	✗
Speculative Generality	Switch Statements	Data class	Message Chains	Inappropriate intimacy	Primitive Obsession

Mentimeter

unused abstractions

✓	✗	✗	✗	✗	✗
Speculative Generality	Refused Bequest	Data class	Message Chains	Comments	Primitive Obsession

a.getB().getC().getD().getE().getF()

✗	✗	✗	✓	✗	✗
Speculative Generality	Refused Bequest	Data class	Message Chains	Comments	Primitive Obsession

two classes calling each other's methods a lot

Speculative Generality	Refused Bequest	Data class	Message Chains	Comments	Inappropriate Intimacy
✗	✗	✗	✗	✗	✓